AFRL-SR-AR-TR-09-0096

REPORT DOCUMENTATION PAGE

1. REPORT DATE *(DD-MM-YYYY)* 10-03-2009	**2. REPORT TYPE** FINAL PERFORMANCE REPORT	**3. DATES COVERED** *(From - To)* 01/03/2006 - 30/11/2008
4. TITLE AND SUBTITLE Hybrid Robust Multi-Objective Evolutionary Optimization Algorithm		**5a. CONTRACT NUMBER** FA9550-06-1-0170
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) George S. Dulikravich, George, S. Department of Mechanical and Materials Engineering		**5d. PROJECT NUMBER**
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Florida International University, College of Engineering and Computing, Room EC 3474, Department of Mechanical and Materials Engineering, EC 3474, 10555 West Flagler Street, Miami, Florida 33174		**8. PERFORMING ORGANIZATION REPORT NUMBER** FIU-AFOSR-212600561-4
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFOSR - Air Force Office of Scientific Research, Optimization and Discrete Math/NL, 875 North Randolph Street, Suite 325, Room 3112, Arlington, VA 22203-1768		**10. SPONSOR/MONITOR'S ACRONYM(S)** AFOSR/NM
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
Distribution A - Approved for Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT

A hybrid robust multi-objective optimization algorithm and accompanying software were developed that:

1. utilize several evolutionary optimization algorithms, a set of rules for automatic switching among these algorithms in order to accelerate the overall convergence and avoid termination in a local minimum,
2. involve development of algorithms for multi-dimensional response surfaces (metamodels) that are fast, accurate and robust by utilizing wavelet-based artificial neural networks, polynomials of radial basis functions, and multi-layer self-adapting maps,
3. involve an algorithm based on Bayesian statistics (using Kalman filters and Monte Carlo Markov chains) that will enhance robustness of the multi-objective optimization algorithm by accounting for uncertainties in the input data and in the accuracy of the evaluation methods for the multiple objective functions.

The hybrid evolutionary multi-objective optimization algorithm was also thoroughly tested on a number of standard test problems with two and three simultaneous objectives where the Pareto surface could be continuous and discontinuous. The hybrid optimizer was programmed in such a way that it can be transportable to any single-processor or parallel processor.

15. SUBJECT TERMS
Optimization; multi-objective optimization; evolutionary optimization; response surface

16. SECURITY CLASSIFICATION OF: UNCLASSIFIED			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** George S. Dulikravich
a. REPORT U	**b. ABSTRACT** U	**c. THIS PAGE** U	UU	37	**19b. TELEPHONE NUMBER** *(include area code)* (305) 348-7016

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

FINAL PERFORMANCE REPORT

Submitted to: **Air Force Office of Scientific Research**

Contract/Grant Title:
Hybrid Robust Multi-Objective Evolutionary Optimization Algorithm

Contract/Grant #: **FA9550-06-1-0170**

Reporting Period: **1 March 2006 to 30 November 2008**

Name of Principal Investigator:
George S. Dulikravich, Ph.D., FAAM, FASME
Professor and Chairman
Department of Mechanical and Materials Engineering
Florida International University
College of Engineering and Computing, Room EC 3474
10555 West Flagler Street
Miami, Florida 33174

(305) 348-7016	**(office phone)**
(305) 348-6007	**(FAX)**
dulikrav@fiu.edu	**(E-mail)**
http://maidroc.fiu.edu	

Final Program Manager:
Donald Hearn, Ph.D.
Program Manager, Optimization and Discrete Math/NL
AFOSR/NM
875 North Randolph Street
Suite 325, Room 3112
Arlington, VA 22203-1768
(703) 696-1142, Fax
(703) 696-8450, DSN 426-8429
Donald.hearn@afosr.af.mil
Interim Program Manager:
Fariba Fahroo, Ph.D.
Program Manager, Computational Mathematics
(and Optimization and Discrete Math Portfolio)
AFOSR/NM
875 North Randolph Street
Suite 325, Room 3112
Arlington, VA 22203
(703) 696-8429, Fax (703) 696-8450, DSN 426-8429
fariba.fahroo@afosr.af.mil
Original Program Manager
Maj. Todd E. Combs, USAF, Ph.D.
Program Manager: Optimization & Discrete Math/NL

Submission Date: **March 10, 2009**

HYBRID ROBUST MULTI-OBJECTIVE EVOLUTIONARY OPTIMIZATION ALGORITHM

EXECUTIVE SUMMARY

Contents **Page**

Statement of Objectives

Currently, a Russian commercially available software named IOSO is the most efficient and the most robust multi-objective optimization software. Similarly, the second best multi-objective optimization software package, modeFrontier, was also developed in a foreign country (Italy) and is licensed by their software company. The only multi-objective optimization software package that was developed in the United States and is licensed by the U.S. software company is Engineous. However, performance speed, robustness and overall sophistication of Engineous optimizer are significantly inferior in comparison to modeFrontier and especially to IOSO performance speed, versatility and robustness. Performance of IOSO, modeFrontier and Engineous multi-objective optimizers published in the open literature clearly indicates that IOSO, which involves concepts of neural networks, radial basis functions, and self-adapting response surface methodologies, requires the minimum number of the objective function evaluations and that it is the most versatile and robust multi-objective optimizer.

Based on our experience in developing hybrid single-objective constrained optimization algorithms, the objective of this effort was to develop a hybrid multi-objective constrained optimization algorithm that will be robust and very fast, thus, emulating some of the main advantages of IOSO algorithm.

The proposed hybrid robust multi-objective optimizer was developed to:

1. utilize several evolutionary optimization algorithms, a set of rules for automatic switching among these algorithms in order to accelerate the overall convergence and avoid termination in a local minimum,
2. involve development of algorithms for multi-dimensional response surfaces (metamodels) that are fast, accurate and robust by utilizing wavelet-based artificial neural networks, polynomials of radial basis functions and self-adapting maps,
3. involve an algorithm based on Bayesian statistics (using Kalman filters, Markov chains, *etc.*) that will enhance robustness of the multi-objective optimization algorithm by accounting for uncertainties in the input data and in the accuracy of the evaluation methods for the multiple objective functions.

The proposed hybrid evolutionary multi-objective optimization algorithm was also to be thoroughly tested on a number of standard test problems with two and three simultaneous objectives where the Pareto surface could be continuous and discontinuous.

The proposed robust hybrid optimizer was to be programmed in such a way that it can be transportable to any single-processor or a parallel processor computing platform.

Summary of Significant Results and Recommendations for Future Research

One of the main conclusions of this research is that any new multi-objective optimization algorithms should involve an efficient and reliable interaction of the following:

- A combination of several multi-objective evolutionary optimization algorithms,
- A fast and accurate algorithm for fitting large dimensionality response surfaces using very small data sets,
- An algorithm that can efficiently and reliably account for statistical noise (uncertainty)

In other words, even the best possible multi-objective evolutionary optimization algorithm for one class of problems cannot be expected to perform better than any other optimization algorithm for all classes of optimization problems. Therefore, a combination of several optimization algorithms should be used in order to assure reliability of the evolutionary multi-objective optimization. Results presented in this report confirm that MOHO is one such optimization concept that works.

Multi-dimensional response surfaces (metamodels) are found to be the key to efficiency and feasibility of any realistic multi-objective optimization where only small data sets are available, because of the very high cost of experimentally or computationally evaluating the objective functions. Among the several methods developed in this project for automatic generation of response surfaces, only those methods that offer consistently high accuracy at a relatively low computing cost are attractive. This is especially becoming critical with the increase in the number of the design variables (dimensions of a response surface) as industry is demanding computationally efficient and accurate response surface algorithms for optimization of realistic problems where numbers of design variables run into hundreds and soon into thousands.

Comparative analysis of the three methods that were developed in this project (wavelet based neural networks, polynomials of radial basis functions, and multi-layer self-assembling concept) clearly points at the following conclusions:

WNN (wavelet based neural networks) should be used only for relatively low dimensional response surface fits where the number of design variables is less than approximately 35. For higher dimensional response surfaces, the accuracy of this algorithm rapidly decreases and the computing costs rapidly increases. Also, this method does not give accurate results for small and scarce data sets.

Polynomials of radial basis functions are very accurate and computationally fast for fitting response surfaces having up to approximately 1000 design variables. For problems having a larger number of design variables (dimensions of the response surface), the matrices that need to be solved become exceedingly ill-conditioned which leads to deterioration of accuracy of the fitting. At the same time, the computing costs for solving these large matrices start increasing rapidly. This method gives good results even for small data sets and reasonable results for scarce data sets.

Response surfaces generation using a hybridized multi-layer self-organizing was developed with a capability to use either linear, quadratic, cubic, or quartic local polynomials at each node of each level of the multi-layer tree. It was found that the most accurate approach is to use a lower order local fitting (typically using quadratic polynomials) followed by a polynomial radial basis function fitting the resulting error. This method for generating multi-dimensional surfaces is quite computationally expensive for lower dimensionality response surfaces. However, this method is the most promising response surface fitting method for very large dimensionality surfaces as its computational cost increases approximately linearly with the number of design variables. This method is capable of giving good results for small and even scarce data sets.

Uncertainty evaluation of initial data and consequently of the output results was developed using Bayesian statistics based on Kalman filters for linear problems and on Markov chains Monte Carlo (MCMC) filters for nonlinear problems. The MCMC method was proven to be significantly more accurate and robust, but it is computationally one order of magnitude costlier.

Multi-Objective Hybrid Optimization (MOHO) With Automatic Switching Among Individual Search Algorithms

The MOHO software [1,2,3] that was developed as a part of this effort is a high level relay hybrid metaheuristic algorithm [4]. In its current version, three different evolutionary multi-objective search algorithms are coordinated and applied in order to expedite the search for a Pareto front. Like many other evolutionary algorithms, MOHO runs in steps of population generations. At each generation the algorithm that is selected makes a new generation using any or all of the information provided to it: the last generation's population and the latest non-dominated set. Then, the MOHO algorithm combines the new generation and the latest non-dominated set to create a new non-dominated set. MOHO keeps track of this process to detect five possible improvements to the dominated set (the Pareto approximation). If the particular search algorithm can achieve at least two of any of the five improvements, this algorithm is allowed to create the next generation. MOHO has been successfully run on Microsoft Windows workstations, Linux workstations and Beowulf style clusters. The software can run in two modes; serial and parallel. In serial mode the optimization and objective function evaluation occur on the same processor. In parallel mode, the optimizer runs on a processor (the cluster server in this case) and the objective function evaluations are then made into jobs that are submitted to the job manager of the cluster in question. This architecture is preferred by the authors to allow for large parallelized computational fluid dynamics and finite element analysis based objective function calls that consume significant amounts of computing time.

MOHO uses three multi-objective optimization algorithms:

1. Strength Pareto Evolutionary Algorithm (SPEA-2) by Zitzler *et al.* [5],
2. Multi-objective implementation [6] of the single objective Particle Swarm (referred to as MOPSO here) by Eberhardt *et al.* [7],
3. Non-Sorting Differential Evolution (NSDE) algorithm which is a low level hybrid metaheuristic search combining NSGA-II by Deb *et al.* [8] and Differential Evolution by Storn and Price [9].

Each of these constituent search algorithms was modified, if needed, to fit into MOHO's system of maintaining the non-dominated set and clustering outside of the search algorithms. Also each search algorithm was set up to be able to accept populations and non-dominated sets in a generalized format to allow the hybridization to run smoothly. These conditions do not adversely affect the application of the individual search algorithms.

SPEA-2 was utilized in the following manner for this work:

1. A population P and a non-dominated set P' are handed to the algorithm from the centralized part of the MOHO software.
2. Then, the fitness is calculated for members of P and P'.
3. Binary tournament selection is used to select the new set of offspring from the mating pool of P+P'.
4. Two-point crossover and bit mutation are performed, but can be changed by the user from the input to better suit a given problem.
5. New population and old non-dominated population are returned to the centralized portion of MOHO where the new P' is generated and clustering is performed, if needed.

The SPEA-2 algorithm does not perform the merging of populations and non-dominating sets. This step is performed by MOHO, external to the search, so the switching algorithm can monitor the progress of the non-dominated set.

The multi-objective Particle Swarm algorithm used for this work is derived from the original Particle Swarm Optimization (PSO) algorithm developed by Eberhardt *et al.* [7]. Some Multi-Objective Particle Swarm Optimization (MOPSO) algorithms utilize weighted sums of PSO algorithms [6]. While this type of application of PSO is effective in its own right, an optimization algorithm solving multi-objective problems in a Pareto optimal sense was needed. To modify PSO for multi-objective optimization, the definition of personal best value and global best values have been modified. In MOPSO, the personal best value for a particle is the non-dominated objective in the particle's search history. The global best value is the member of the current generation's non-dominated set that is closest (in objective space) to the particle for which the velocity is being calculated. Other than the method for choosing the global and personal best for each particle, the rest of MOPSO remains true to the original PSO. The MOPSO algorithm was utilized in the following way for this work:

1. A population P and a non-dominated set P' are handed to the algorithm from the centralized part of the MOHO software.
2. A velocity vector for each particle is calculated using the technique described earlier.
3. The displacement for each particle is calculated using the equations of motion and unit time step.
4. New population and old non-dominated population are returned to the centralized portion of MOHO where the new P' is generated and clustering is performed, if needed.

The NSDE created for this work is a low level combination of NSGA-II [8] and differential evolution (DE) [9]. In particular, the mutation operator from DE replaces the mutation operator in the original NSGA-II. The rest of the algorithm is from NSGA-II. At the beginning of the algorithm, the last population and the non-dominated population are used to play the roles of the offspring and parents from the last generation, respectively. At this point, NSDE continues on like NSGA-II until the new generation is created. Then, the new population and the old non-dominated set are handed back to the switching algorithm in MOHO.

After each generation is created, and the new non-dominated set is identified, the software assigns the new non-dominated set a score from zero to five. If the new non-dominated set gets a score of two or better, the algorithm that generated the set was allowed to create the next generation of population points. An algorithm scores a point for each of five possible improvements that are achieved from one generation to the next. Also, if an algorithm has run consecutively beyond a user specified iteration (generation) limit, the run switches to the next constituent search algorithm determined by the algorithm pointer array. This rule was used to allow all constituent search algorithms an opportunity to improve the Pareto approximation.

The switching algorithm compares the non-dominated set from the current generation to the non-dominated set of the previous generation. The comparison process consists of looking at five desired improvements to the Pareto approximation. The improvements are actually gains in five performance criteria (quality factors). More than one quality factor was used because of the work of Zitzler *et al.* [10, 11], where it is shown that as opposed to the situation in a single objective optimization, one quality factor alone should not be used to compare two Pareto approximations. This is why an algorithm must score a minimum of two in order to continue creating new generations of population points. This work also puts forth definitions of compatibility and completeness for quality factors. After the improvement metrics for the MOHO algorithm are presented, the applicability of the compatibility and completeness presented in Zitzler *et al.* [10, 11] will be discussed.

The main part of MOHO handles combining the new generation with the most recent non-dominated set to form the new non-dominated set by employing objective space based clustering as needed and calculating the improvements. The clustering is employed only as a non-dominated set trimming tool when the software determines that the non-dominated set will grow beyond the user defined limit. Five improvements were developed and used in this work because there is a concern that adding too many Pareto based calculations would add considerable overhead to the software. Now that the algorithm switching logic has been discussed, the five improvements will be defined.

Improvement #1 – Non-Dominated Set Size Changes

For this improvement, the size of the non-dominated set changes. This change can be either the non-dominated set getting larger or smaller. The non-dominated set grows in size when a new point on the Pareto approximation front is discovered and the old non-dominated set is below the user defined non-dominated set size limit. Shrinking of the non-dominated set occurs when a new point or points, dominate larger set of points from the old non-dominated set. This indicates one of two things: a) the new point(s) significantly redefine the geometry of the non-dominated set, or b) clustering has yet to be employed on the Pareto approximation and multiple points were clustered around each other.

Improvement #2 – A Point from the New Generation Dominates

This improvement is satisfied if any population member of the new generation dominates any member in the last generation's non-dominated set. Any opportunity to improve the Pareto approximation by removing a dominated point is considered an improvement. This can be expressed by:

$$\begin{aligned}
&i)\ \text{Set A} = \text{False}\\
&ii)\ \text{Let m} = \text{Sizeof(Pop) and n} = \text{Sizeof(NonDom)}\\
&iii)\ \left(\text{POP}_{\text{j}} \succ \text{NonDom}_{\text{k}}\ \text{? True: False}\right) \vee A; j = 1,\ldots,m; k = 1,\ldots,n\\
&iv)\ \text{2nd Improvement} = \text{A}
\end{aligned} \tag{1}$$

Improvement #3 – Change in the Dominated Hyper Volume

The hyper volume quality factor has its roots in the hyper volume calculation presented by Deb [12]. When the first population generation is created using Sobol's quasi random sequence generator [13, 14], the worst objective value for each objective from the entire population is collected into a worst case objective vector. This worst case objective vector is used as the common diagonal for all hyper volume calculations in the search; a static datum for the entire run. The improvement is considered fulfilled when the new generation's contribution to the non-dominated set causes a change in the dominated hyper volume for the non-dominated set.

Improvement #4 – Average Distance Change

In this calculation the average Euclidian distance of all the objective vectors of the new generation's non-dominated set is calculated. This is basically the magnitude of the objective vectors, because the datum is the origin of the objective space. If the measure changes from the value of the old generation, this improvement has been met. This improvement tries to capture changes in the geometry of the non-dominated set.

Improvement #5 – Spread of the Don-dominated Set

The equation for the spread is found in a book by Deb [12] and its origins are attributed to Zitzler. This improvement is fulfilled if the new generation's non-dominated set increases the spread over that of the last generation. The equation for the spread is as follows:

$$SPRD = \sqrt{\sum_{m=1}^{M}\left(\max_{i=1}^{|Q|} \text{f}_{\text{m}}^{\text{i}} - min_{i=1}^{|Q|} \text{f}_{\text{m}}^{\text{i}}\right)^2} \tag{2}$$

Improvements #3 and #5 are common metrics for trying to determine if one Pareto approximation set is better than another. The other improvements are not common, since it would be useless to compare final Pareto results from two different multi-objective algorithms.

If the search algorithm in question is not able to achieve at least two improvements, or it has consecutively run for the user specified limiting number of iterations, the latest population and non-dominated set are passed to the next search algorithm. MOHO runs until the maximum number of function evaluations is performed. All optimization run parameters are specified by the user in an external (to the software) input file. MOHO was tested in order to evaluate its capability by running it on test problems from three previously published works. The first work examined is the well known multi-objective optimization comparison paper by Zitzler, Thiele and Deb [10] (referred to as ZDT from this point forward). A copy of the original data from the ZDT paper can be found on the website cited in that paper. In the present work, we will present this data again and inject the results from our MOHO into this original comparison. The second and third works that MOHO will be compared with, are related to each other. In the original NSGA-II paper, Deb *et al.* [8, 10] revisit some of the ZDT problems, present results for some other unconstrained problems, and use NSGA-II to solve some constrained problems. In the present work, our MOHO will was used to solve the unconstrained problems of the original NSGA-II paper. Results from that paper will be compared to MOHO results. Finally, the paper by Vrugt and Robinson compares their AMALGAM [15] algorithm to the performance of NSGA-II for some unconstrained multi-objective optimization test problems. These results will also be included in this work and compared to results from MOHO using the same experimental conditions.

In the ZDT work, the Strength Pareto Evolutionary Algorithm (SPEA), Non-Sorting Genetic Algorithm (NSGA), Vector Evaluated Genetic Algorithm (VEGA), Niched Pareto Genetic Algorithm (NPGA), Hajela and Lin's evolutionary algorithm (HLGA), Fonseca and Fleming's evolutionary algorithm (FFGA), and a Single Objective Evolutionary Algorithm (SOEA) were studied. The algorithms were applied to six multi-objective problems designed by those authors specifically for that work. All test problems represent two-objective optimizations where both objectives are to be minimized. All seven algorithms were compared in two ways.

The optimizer utilizes several multi-objective evolutionary optimization algorithms and orchestrates the application of these algorithms to multi-objective optimization problems, using an automatic internal switching algorithm. The switching algorithm is designed to favor those search algorithms that quickly improve the Pareto approximation and grades improvements using five criteria. A thorough testing of reliability and accuracy of MOHO against a number of prominent multi-objective optimization algorithms and one hybrid optimizer confirmed that MOHO performs reliably and accurately.

For the first results comparing MOHO to the evolutionary multi-objective optimization algorithms from the ZDT work [10, 11, 12], it has been shown that MOHO can outperform the classic evolutionary algorithms for all test problems except ZDT5. Difficulties surrounding ZDT5 test case are not trivial as was confirmed by Deb *et al.* [12] who tested all the ZDT test problems, except ZDT5.

First, a plot of the non-dominated sets for each algorithm was made. To allow for performance fluctuations caused by the random number generators driving the initial population and genetic operators, all algorithms were run 30 times on each of the six problems. The non-dominated plot is generated by making a union of the non-dominated sets for the first five runs, of each algorithm, on each problem. The non-dominated set of the unions is then plotted. In the figures presented here, the results for MOHO are injected into the plot results from ZDT and the random number data from the original plots is removed to provide a clearer view of the performances of the search algorithms. Figures 1 through 6 are the plots for ZDT test problems 1 through 6, respectively, using the original ZDT paper data and MOHO data run for this work.

From these figures it is evident that our MOHO algorithm performs very well on almost all of the ZDT test cases as far as accuracy is concerned. Besides being an accurate evolutionary search algorithm, these figures demonstrate that MOHO is also a reliable algorithm that consistently gives good results.

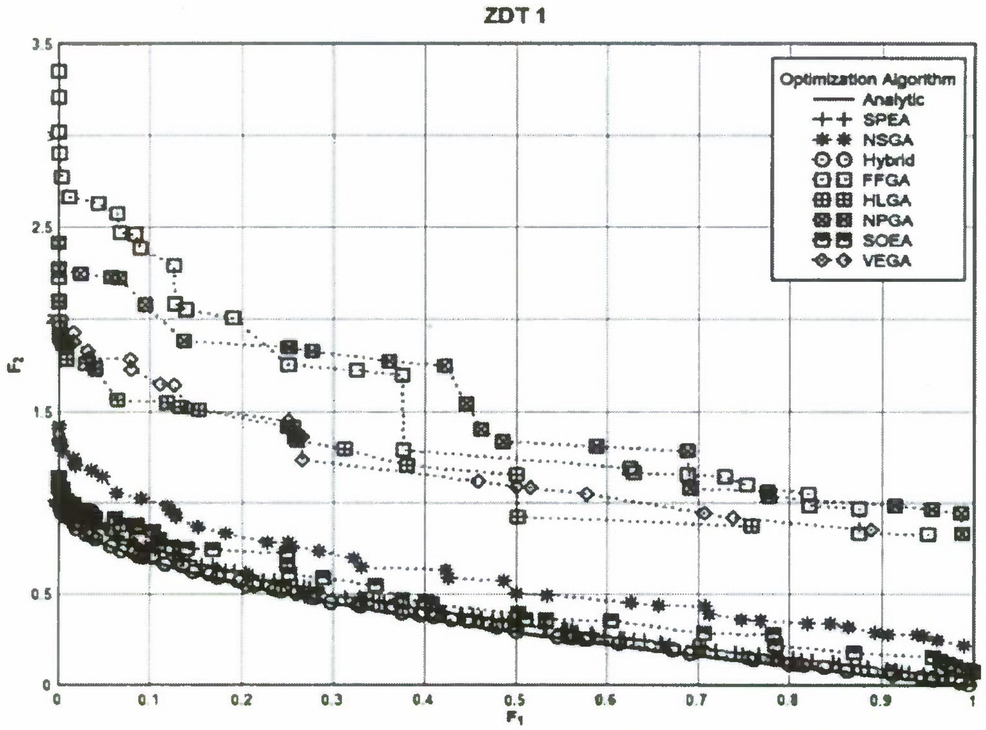


Fig. 1 Results for test problem 1 from ZDT with MOHO results added (circles).

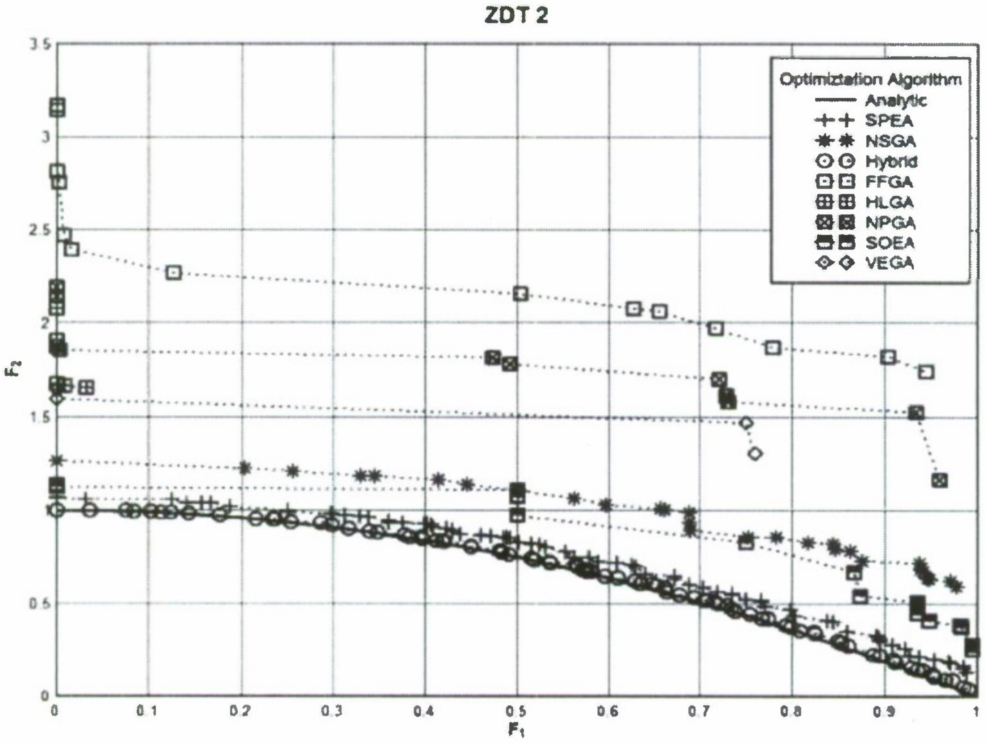


Fig. 2 Results for test problem 2 from ZDT with MOHO results added (circles).

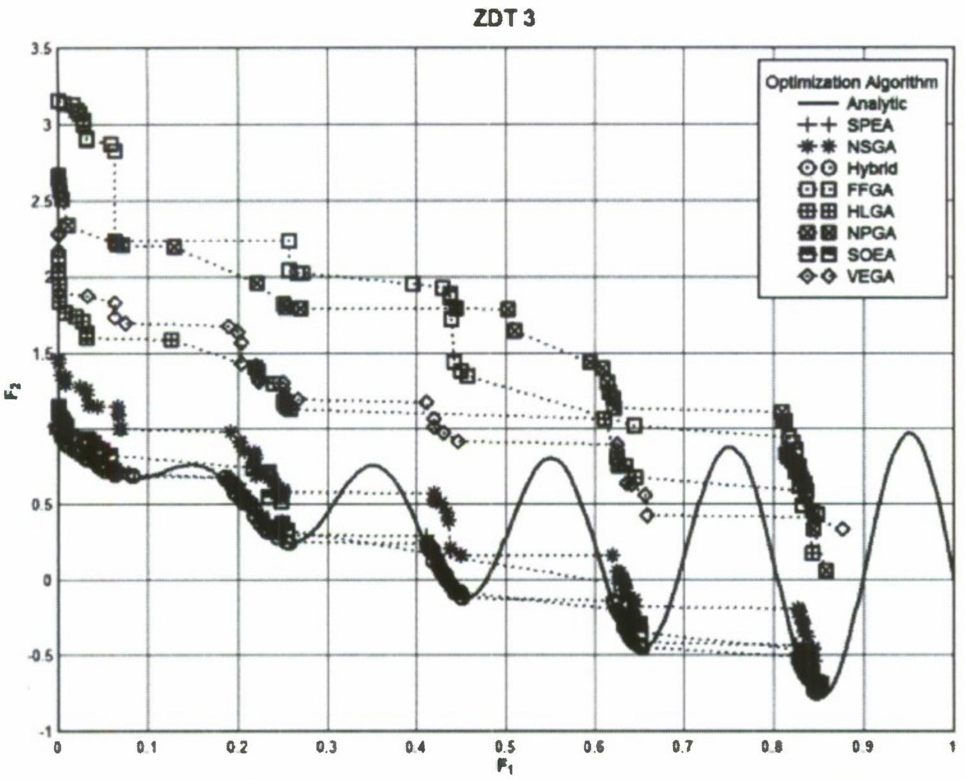


Fig. 3 Results for test problem 3 from ZDT with MOHO results added (circles).

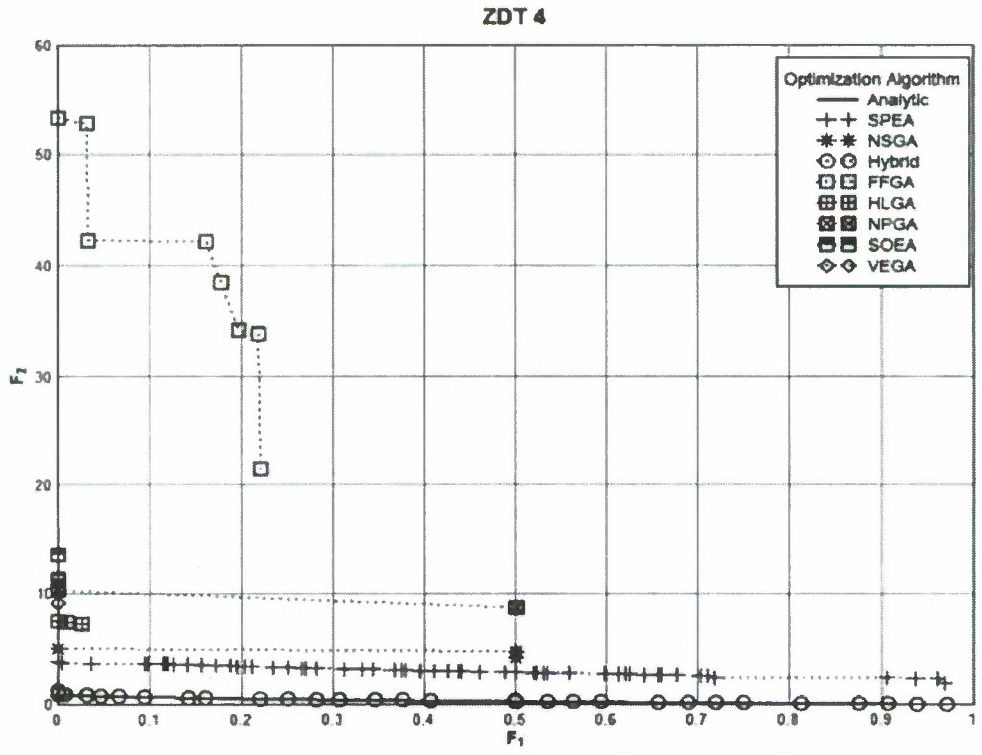


Fig. 4 Results for test problem 4 from ZDT with MOHO results added (circles).

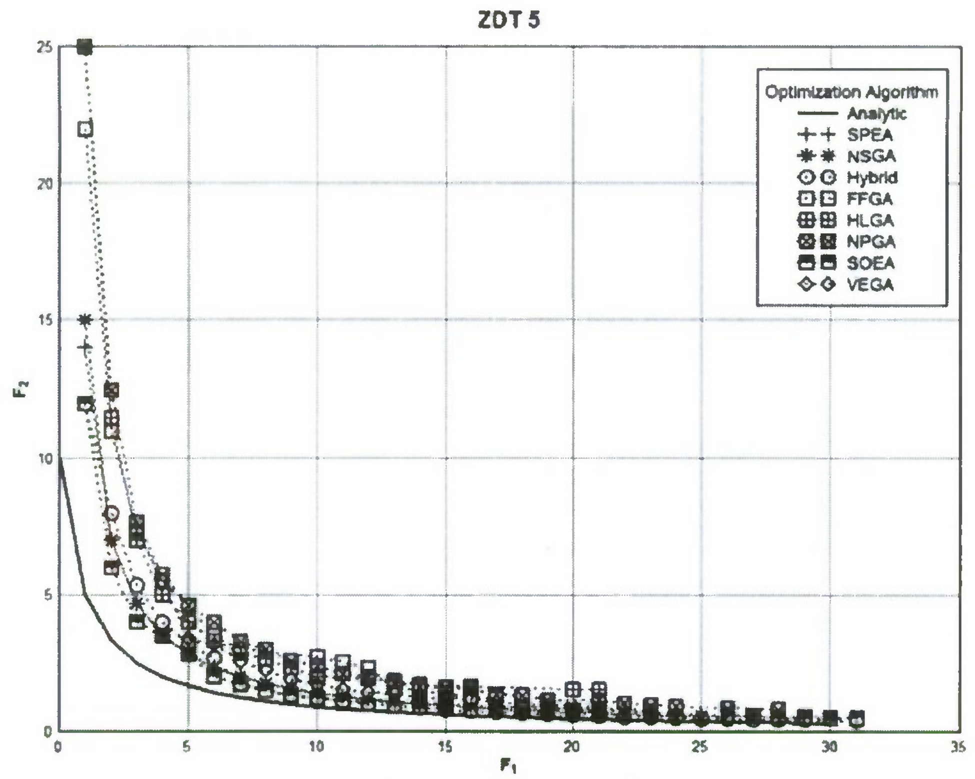


Fig. 5 Results for test problem 5 from ZDT with MOHO results added (circles).

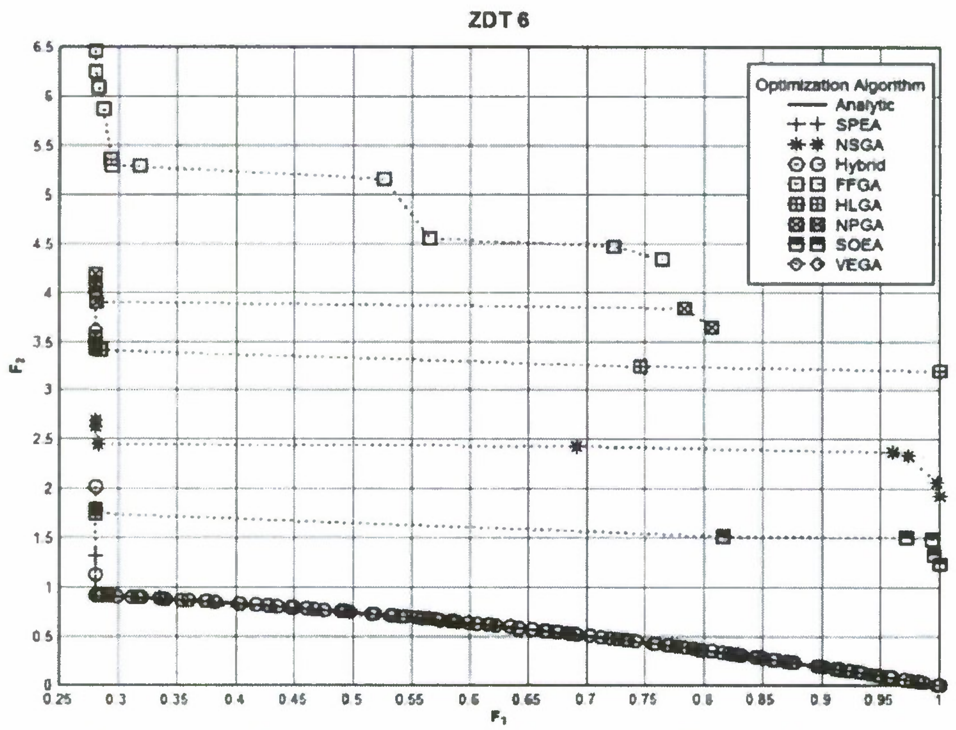


Fig. 6 Results for test problem 6 from ZDT with MOHO results added (circles).

The second comparison mechanism is the "cover" function proposed in the ZDT work. The cover function evaluates what fraction of the non-dominated set of algorithm A is either equal to or weakly dominates the non-dominated set of algorithm B. The formula for the cover functions, as shown in the ZDT work, is:

$$C(A,B) := \frac{\left|\{a'' \in B; \exists a' \in A : a' \underline{p}\, a''\}\right|}{|B|} \quad (3)$$

Of note is the fact that C(A,B) is not necessarily equal to C(B,A).

Figure 7 shows the results for the cover function analysis for all test problems and optimization algorithms compared [3]. In this analysis, all 30 runs of each algorithm are utilized. Each cell in Figure 7 is made by treating the algorithms in the row as algorithm A in equation 3. Each of the 30 Pareto approximations for algorithm A is compared to the algorithm B (column). So, for each Pareto approximation of algorithm A, comparisons with 30 algorithm B Pareto sets are performed.. This means that 30 cover function values are generated. Each column in the box plot (Figure 7) represents the results for one of the ZDT problems. So, each column in the box plot represents a total of 900 Pareto approximation set comparisons.

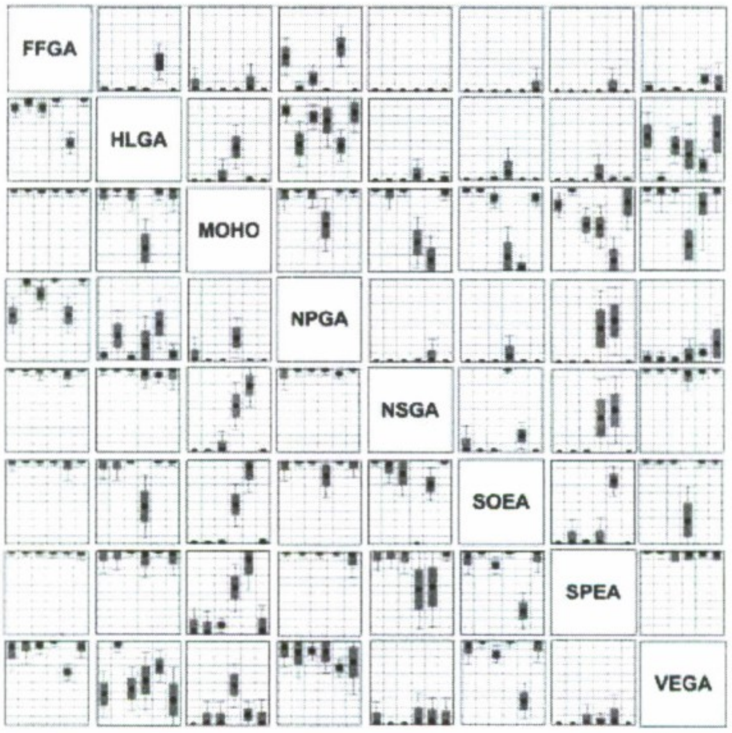


Fig. 7 Cover function analysis from ZDT with MOHO data inserted into the original analysis. As in the original analysis the value at the bottom of the graph is 0 and the value is 1 at the top. The black boxes in the box plots show the average cover value for all 30 runs. The shaded boxes and the whiskers display the one-standard-deviation spread and two-standard-deviation spreads, respectively. (i.e., ±0.5 standard deviations and ±1 standard deviation). The plots are read left to right: the leftmost box shows the results for ZDT1 and rightmost box shows the results for ZDT6.

Using the data generated from solving the ZDT test problems it was possible to perform another analysis of the MOHO algorithm's performance. Figure 8 shows the average percent of total number of function evaluations used by each of the three constituent search algorithms in MOHO when applied to a given ZDT test problem. Figure 8 takes into account all 30 runs used for the ZDT data comparison.

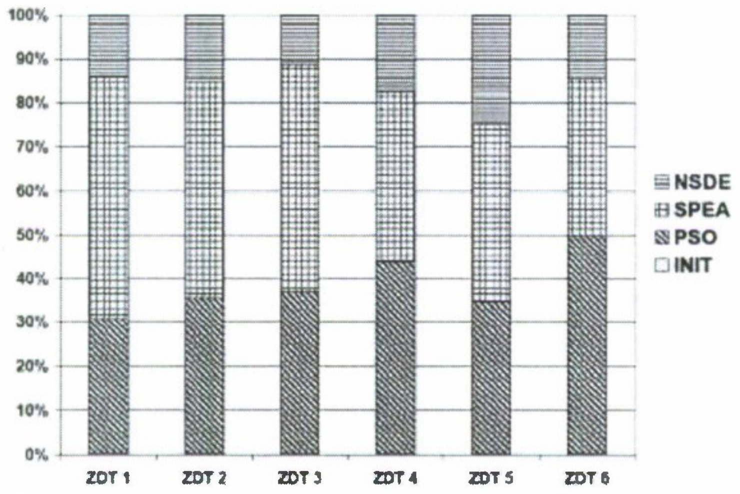


Fig. 8 Average percent of total execution time that each constituent optimization algorithm was used in MOHO for each of the six ZDT test problems. The average is taken over 30 runs used in the ZDT comparison.

Two-Objective Hybrid Optimizer with a Response Surface

Besides MOHO algorithm, we have also developed an entirely different multi-objective hybrid optimization algorithm that currently works only for two-objective problems [16-19]. The general schematic of this hybrid optimizer is given in Figure 9 and Figure 10.

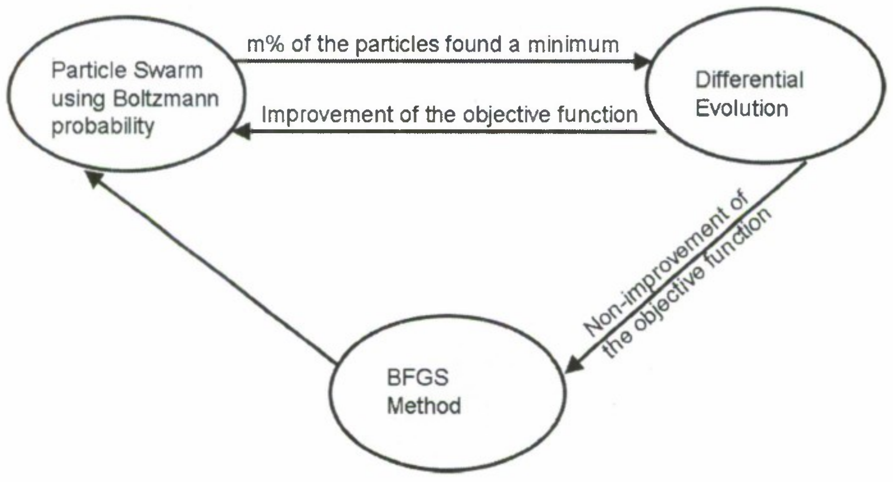

m% of the particles found a minimum
Particle Swarm using Boltzmann probability
Differential Evolution
Improvement of the objective function
Non-improvement of the objective function
BFGS Method

Figure 9 – Global procedure for the hybrid optimization method ***H1***

The method starts with a randomly generated population matrix **P** in the domain of interest.

Thus, successive combinations of chromosomes and mutations are performed, creating new generations until an optimum value is found. The iterative process is given by

$$\mathbf{x}_i^{k+1} = \delta_1 \mathbf{x}_i^k + \delta_2 \left[\boldsymbol{\alpha} + F(\boldsymbol{\beta} - \boldsymbol{\gamma}) \right] \tag{4}$$

where

- $\mathbf{x}_i$ is the *i*-th individual of the vector of parameters.
- α, β and γ are three members of population matrix **P**, randomly chosen.
- F is a weight function, which defines the mutation ($0.5 < F < 1$).
- k is a counter for the generations.
- δ_1 and δ_2 are delta Dirac functions that define the mutation.

In this minimization process, if $f(\mathbf{x}^{k+1}) < f(\mathbf{x}^k)$, then $\mathbf{x}^{k+1}$ replaces $\mathbf{x}^k$ in the population matrix **P**. Otherwise, $\mathbf{x}^k$ is kept in the population matrix.

The binomial crossover is given as

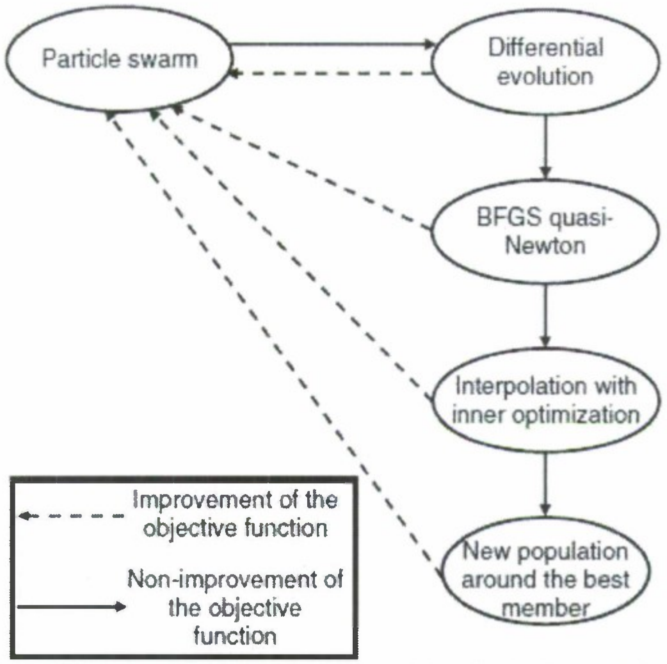
$$\begin{aligned} \delta_1 = 0, &\text{ if } \mathrm{R} < \mathrm{CR} \\ 1, &\text{ if } \mathrm{R} > \mathrm{CR} \end{aligned} \tag{5}$$

where CR is a factor that defines the crossover ($0.5 < \mathrm{CR} < 1$) and R is a random number with uniform distribution between 0 and 1.

In the hybrid optimizer ***H1***, when a certain percent of the particles find a minimum, the algorithm switches automatically to the differential evolution method and the particles are forced to breed. If there is an improvement in the objective function, the algorithm returns to the particle swarm method, meaning that some other region is more prone to having a global minimum. If there is no improvement on the objective function, this can indicate that this region already contains the global value expected and the algorithm automatically switches to the BFGS method in order to find its location more precisely. In Figure 9, the algorithm returns to the particle swarm method in order to check if there are no changes in this location and the entire procedure repeats itself. After some maximum number of iterations is performed (e.g., five) the process stops.

The hybrid optimizer ***H2*** [19] is quite similar to the ***H1***, except by the fact that is uses a response surface method at some point of the optimization task. The global procedure is illustrated in Fig. 10. It can be seen from this figure that after a certain number of objective functions were calculated, all this information was used to obtain a response surface. Such a response surface is then optimized using the same proposed hybrid code defined in the ***H1*** optimizer so that it fits the calculated values of the objective function as closely as possible. New values of the objective function are then obtained very cheaply by interpolating their values from the response surface.

In Figure 10, if the BFGS cannot find any better solution, the algorithm uses a radial basis function interpolation scheme to obtain a response surface and then optimizes such response surface using the same hybrid algorithm proposed. When the minimum value of this response surface is found, the algorithm checks to see if it is also a solution of the original problem. Then, if there is no improvement of the objective function, the entire population is eliminated and a new population is generated around the best value obtained so far. The algorithm returns to the particle swarm method in order to check if there are no changes in this location and the entire procedure repeats itself. After a specified maximum number of iterations is performed (*e.g.*, five) the process stops.


Particle swarm
Differential evolution
BFGS quasi-Newton
Interpolation with inner optimization
New population around the best member
Improvement of the objective function
Non-improvement of the objective function
Figure 10 – Global procedure for the hybrid optimization method ***H2***

The new algorithm, which will be called ***H3,*** is an extension of the previous ones. The global procedure is outlined below:

1. Generate an initial population, using the real function (not the interpolated one) $f(\mathbf{x})$. Call this population $\mathbf{P}_{real}$.
2. Determine the individual that has the minimum value of the objective function, over the entire population $\mathbf{P}_{real}$ and call this individual $\mathbf{x}_{best}$.
3. Determine the individual that is more distant from the $\mathbf{x}_{best}$, over the entire population $\mathbf{P}_{real}$. Call this individual $\mathbf{x}_{far}$.
4. Generate a response surface, with the methodology at section 2, using the entire population $\mathbf{P}_{real}$ as training points. Call this function $g(\mathbf{x})$.
5. Optimize the interpolated function $g(\mathbf{x})$ using the hybrid optimizer ***H1***, defined above, and call the optimum variable of the interpolated function as $\mathbf{x}_{int}$. During the generation of the internal population to be used in the ***H1*** optimizer, consider the upper and lower bounds limits as the minimum and maximum values of the population $\mathbf{P}_{real}$ in order to not extrapolate the response surface.
6. If the real objective function $f(\mathbf{x}_{int})$ is better than all objective function of the population $\mathbf{P}_{real}$, replace $\mathbf{x}_{far}$ by $\mathbf{x}_{int}$. Else, generate a new individual, using the Sobol pseudo-random sequence generator within the upper and lower bounds of the variables, and replace $\mathbf{x}_{far}$ by this new individual.
7. If the optimum is achieved, stop the procedure. Else, return to step 2.

From the sequence above, one can notice that the number of times that the real objective function $f(\mathbf{x})$ is called is very small. Also, from step 6, one can see that the space of search is reduced at each iteration. When the response surface $g(\mathbf{x})$ is no longer capable to find a minimum, a new call to the real function $f(\mathbf{x})$ is made to generate a new point to be included in the interpolation. Since the CPU time to calculate the interpolated function is very short, the maximum number of iterations of the ***H1*** optimizer can be very large (e.g., 1000 iterations).

The hybrid optimizer ***H3*** was compared against the optimizer ***H1***, ***H2*** and the commercial code IOSO 2.0 for some standard test functions. The first test function was the Levy #9 function [20], which has 625 local minima and 4 variables. Such function is defined as

$$f(\mathbf{x}) = \sin^2(\pi - z_1) + \sum_{i=1}^{n-1}(z_i - 1)^2\left[1 + 10\sin^2(\pi z_{i+1})\right] + (z_4 - 1)^2 \tag{6}$$

where

$$z_i = 1 + \frac{x_i - 1}{4}, (i = 1,4) \tag{7}$$

The function is defined within the interval -10 ≤ **x** ≤ 10 and its minimum is *f*(**x**) = 0 for **x** = 1. Figure 11 shows the optimization history of the IOSO, ***H1***, ***H2*** and ***H3*** optimizers. Since the ***H1***, ***H2*** and ***H3*** optimizers are based on random number generators (because of the Particle Swarm module), we present the best and worst estimates for these three optimizers.

From Fig. 11, it can be seen that the performance of the ***H3*** optimizer is very close to the IOSO commercial code. The ***H1*** code is the worst and the ***H2*** optimizer also has a reasonable good performance. It is interesting to note that the ***H1*** code is the only one that doesn't have a response surface model implemented.

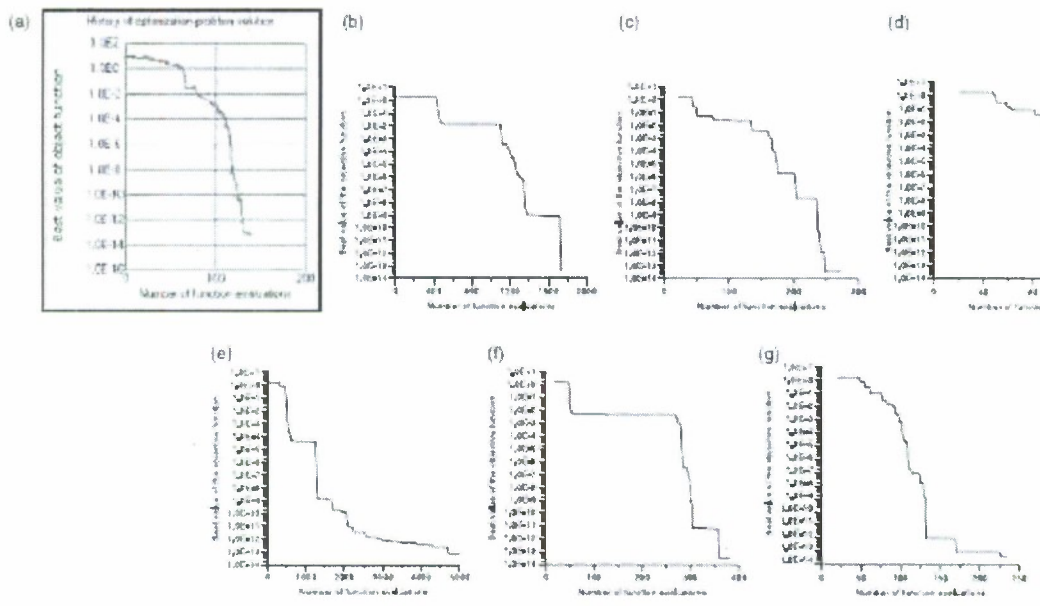


Figure 11 – Optimization history of the Levy #9 function for the (a)IOSO, (b)***H1-best***, (c)***H2-best***, (d)***H3-best***, (e)***H1-worst***, (f)***H2-worst*** and (g)***H3-worst*** optimizers

The second function tested was the Griewank function [20], defined as

$$f(\mathbf{x}) = \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \qquad x_i \in \left]-600, 600\right[\quad ,(i = 1,2) \tag{8}$$

The global minima for this function is located at **x** = 0 and is $f(\mathbf{x}) = 0$. The function has an incredible number of local minima, making the optimization task quite difficult.

Figure 12 shows the optimization history of the IOSO commercially available Russian optimization code IOSO [21], ***H1***, ***H2*** and ***H3*** optimizers. Again, the best and worst results for ***H1***, ***H2*** and ***H3*** are presented. From this figure, it is clear that the ***H1***, ***H2*** and ***H3*** optimizers are much better than the IOSO commercial code. The ***H1*** code was the best, while the ***H2*** sometimes stopped at some local minima. The worst result of the ***H3*** optimizer was, however, better than the result obtained by IOSO. It is worth to notice that the, with more iterations, the ***H3*** code could reach the minimum of the objective function, even for the worst result.

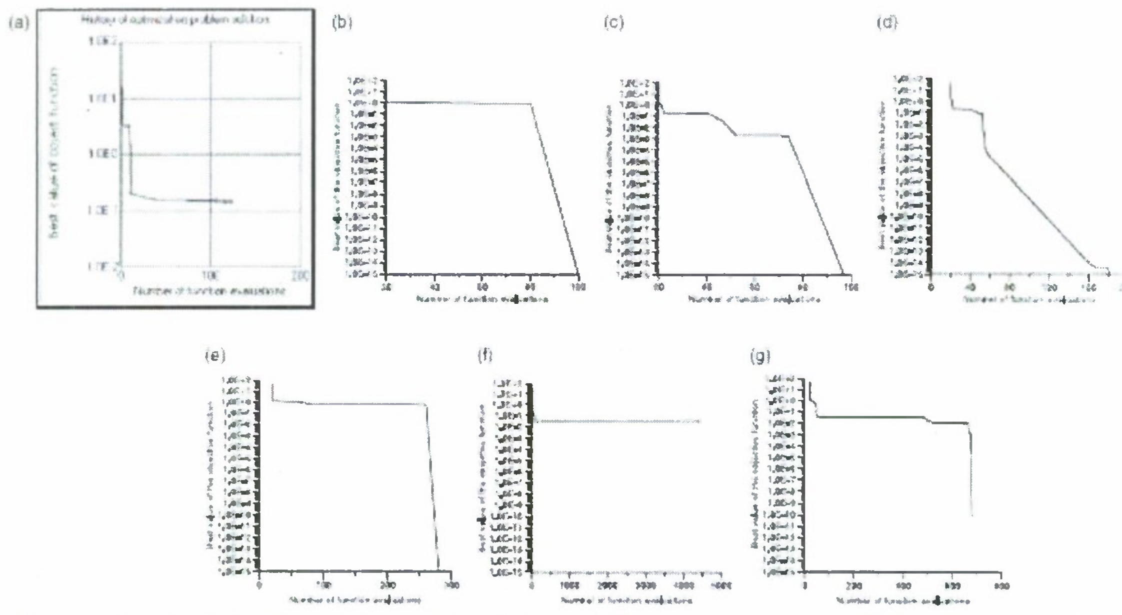


Figure 12 - Optimization history of the Griewank function for the (a)IOSO, (b)***H1-best***, (c)***H2-best***, (d)***H3-best***, (e)***H1-worst***, (f)***H2-worst*** and (g)***H3-worst*** optimizers

The next test function implemented was the Rosenbrook function [22], which is defined as

$$f(x_1, x_2) = 100\left(x_2 - x_1^2\right)^2 + \left(1 - x_1\right)^2 \tag{9}$$

The function is defined within the interval $-10 \leq \mathbf{x} \leq 10$ and its minimum is $f(\mathbf{x}) = 0$ for **x** = 1. Figure 13 shows the optimization history of the IOSO, ***H1***, ***H2*** and ***H3*** optimizers.

For this test function, which is almost flat close to the global minima, the IOSO code was the one with the best performance, followed by the ***H3*** optimizer. The ***H2*** did not perform well and the ***H1*** was able to get close to the minimum, but with a huge number of objective function calculations. When looking at the ***H3*** results, the final value of the objective function differs by some orders of magnitude. However, the optimum solution obtained with this new optimizer was $x_1 = 0.9996$ and $x_2 = 0.9992$, while the IOSO obtained $x_1 = 1.0000$ and $x_2 = 1.0000$. Thus the relative error among the variables was less than 0.01%, indicating that despite of the discrepancy among the final value of the objective function, the ***H3*** code was able to recover the value of the optimum variables with a negligible relative error.

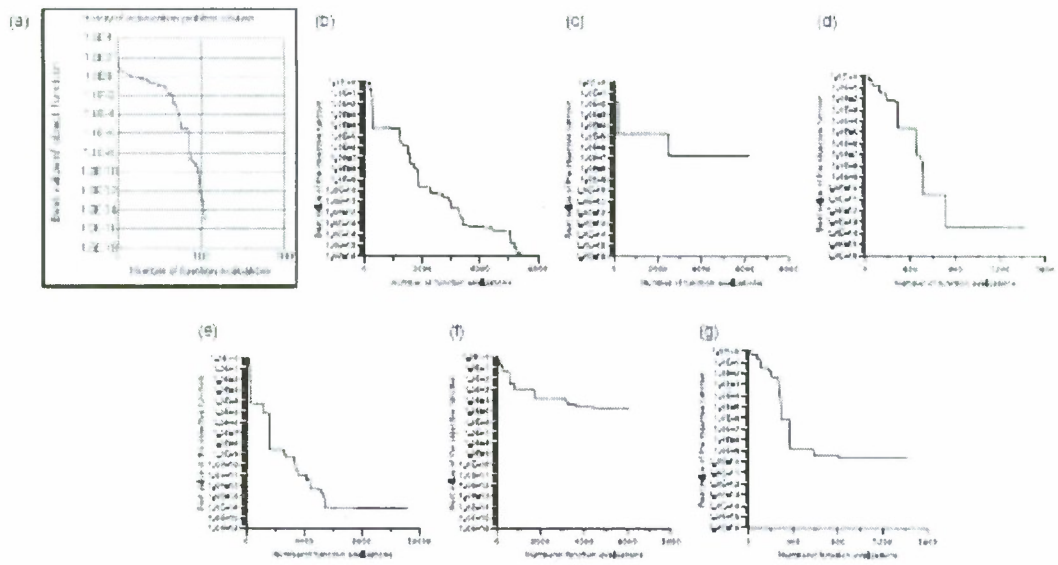


Figure 13 – Optimization history of the Rosenbrook function for the (a)IOSO, (b)***H1-best***, (c)***H2-best***, (d)***H3-best***, (e)***H1-worst***, (f)***H2-worst*** and (g)***H3-worst*** optimizers

The last test function analyzed was the Mielle-Cantrel function [23], which is defined as

$$f(\mathbf{x}) = \left[\exp^{(x_1 - x_2)}\right]^4 + 100(x_2 - x_3)^6 + \arctan^4(x_3 - x_4) + x_1^2 \tag{10}$$

The function is defined within the interval $-10 \le \mathbf{x} \le 10$ and its minimum is $f(\mathbf{x}) = 0$ for $x_1 = 0$ and $x_2 = x_3 = x_4 = 1$. Figure 14 shows the optimization history of the IOSO, ***H1***, ***H2*** and ***H3*** optimizers. Again, the best and worst results for ***H1***, ***H2*** and ***H3*** are presented.

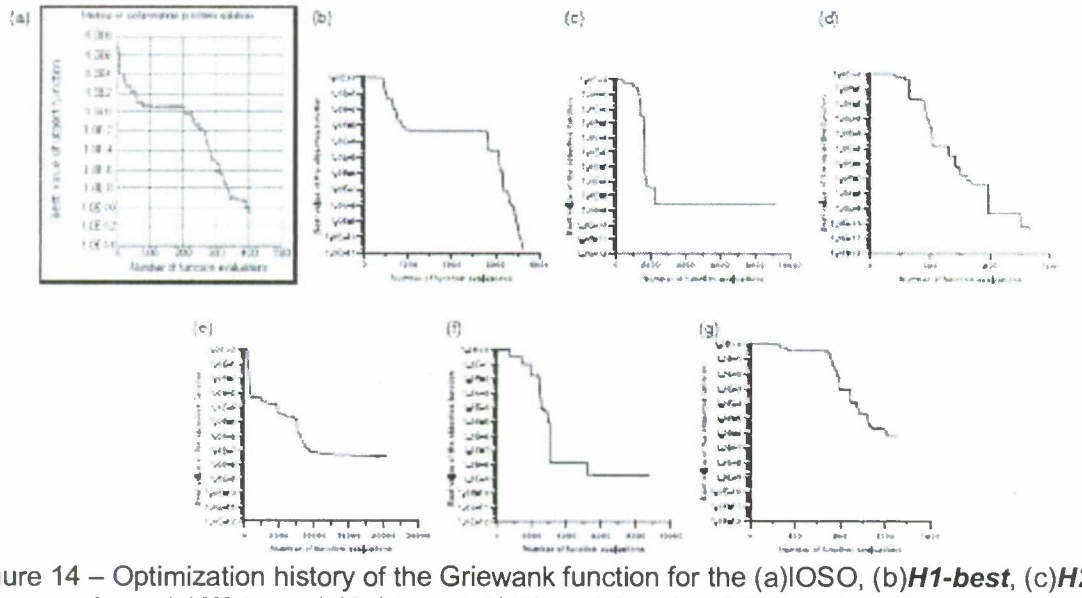


Figure 14 – Optimization history of the Griewank function for the (a)IOSO, (b)***H1-best***, (c)***H2-best***, (d)***H3-best***, (e)***H1-worst***, (f)***H2-worst*** and (g)***H3-worst*** optimizers

For this function, the IOSO code was the best, followed by the ***H3***. The ***H2*** code performed very bad again the the ***H1*** was able to get to the global mininum after a huge numbe of objective function calculations. As occurred with the Rosenbrook function, in spite of the ***H3*** result for the objective function differ from the IOSO code, the final values of the variables were x_1=4.0981x10^{-8}, x_2=0.9864, x_3=0.9688 and x_4=0.9626 for the ***H3*** optimizer and x_1=-0.1216x10^{-5}, x_2=1.002, x_3=0.9957 and x_4=0.9962 for the IOSO code.

ALGORITHM DEVELOPMENTS FOR RESPONSE SURFACES

Three distinctly different algorithms have been developed for automatic generation of multi-dimensional response surfaces (metamodels) supported by a small number of high fidelity values of the objective functions. These three methods were based, respectively, on:
(1) Wavelet based neural networks [24]
(2) Polynomial radial basis functions [25-28]
(3) Self-organizing maps and graph theory [35]

Response Surfaces using Wavelet-Based Neural Networks (WNN)

This is an alternative technique to search the proper activation functions for the construction of WNN that helps to build response surface for predicting multi-dimensional function spaces accurately and efficiently. Several modifications to the architecture and basis function in WNN were suggested and tested using diverse test functions. Simulation results show that WNN for fitting multi-dimensional surfaces having moderate number of dimensions (<35) can be implemented accurately. The modified WNN developed in such a way had about 5 – 7 % average absolute error on training data in most of the test problems. The final testing for most of the test functions was done using 1000 – 1200 data points. It was found that about ten to twelve activation nodes in the hidden layer of WNN were adequate for good predictions, that is, 5% average absolute error on training data. Further addition of activation nodes in WNN improved the accuracy on training data only slightly, but did not help in improving the accuracy on testing data further [24]. This implies that for the given dataset, a WNN with about ten to twelve activation nodes extracted most of the information regarding the topology of the functional space. Finally, a hybrid WNN network was developed using the concept of having multiple WNNs and helped in increasing the accuracy of prediction of highly non-linear functions. From Figure 15 it appears that WNN is appropriate for low-dimensional response surfaces.

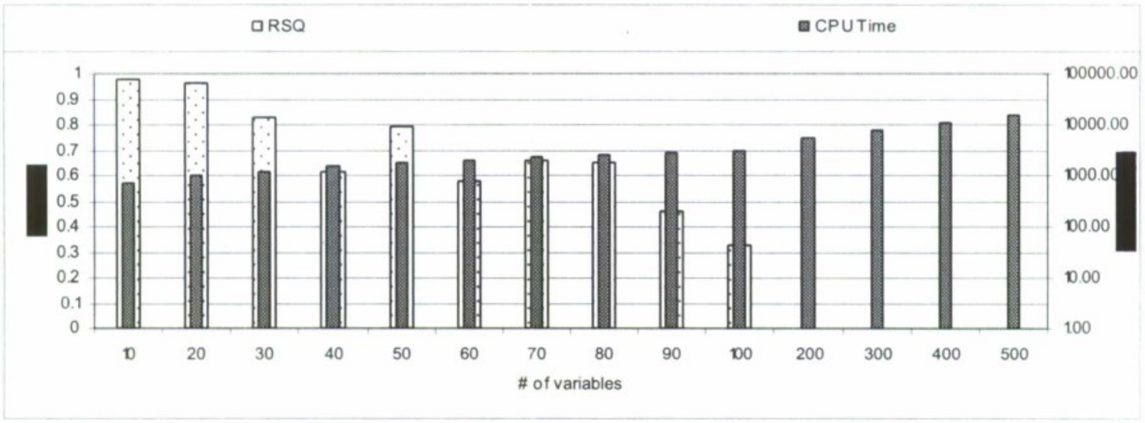


Figure 15 – R2 accuracy results and CPU time for a WNN with one subnet.

Response Surfaces using Polynomials of Radial Basis Functions

Let us suppose that we have a function of L variables x_i, $i = 1,\ldots, L$. The RBF model used in this work has the following form [25-28]

$$s(\mathbf{x}) = f(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j \phi\left(\left|\mathbf{x}-\mathbf{x}_j\right|\right) + \sum_{k=1}^{M}\sum_{i=1}^{L} \beta_{i,k} p_k(x_i) + \beta_0 \tag{11}$$

where $\mathbf{x}=\{x_1,\ldots,x_i,\ldots,x_L)$ and $f(\mathbf{x})$ is known for a series of points $\mathbf{x}$. Here, $p_k(x_i)$ is one of the M terms of a given basis of polynomials [29]. This approximation is solved for the α_j and $\beta_{i,k}$ unknowns from the system of N linear equations, subject

$$\begin{gathered}\sum_{j=1}^{N} \alpha_j p_k(x_i) = 0 \\ \mathrm{M} \\ \sum_{j=1}^{N} \alpha_j p_k(x_L) = 0\end{gathered} \tag{12}$$

$$\sum_{j=1}^{N} \alpha_j = 0 \tag{13}$$

In this work, the polynomial part of Eq. (11) was taken as

$$p_k(x_i) = x_i^k \tag{14}$$

and the radial basis functions are selected among the following [30]

$$\text{Multiquadrics:} \quad \phi\left(\left|x_i - x_j\right|\right) = \sqrt{\left(x_i - x_j\right)^2 + c_j^2} \tag{15}$$

$$\text{Gaussian:} \quad \phi\left(\left|x_i - x_j\right|\right) = \exp\left[-c_j^2\left(x_i - x_j\right)^2\right] \tag{16}$$

$$\text{Squared multiquadrics:} \quad \phi\left(\left|x_i - x_j\right|\right) = \left(x_i - x_j\right)^2 + c_j^2 \tag{17}$$

$$\text{Cubical multiquadrics:} \quad \phi\left(\left|x_i - x_j\right|\right) = \left[\sqrt{\left(x_i - x_j\right)^2 + c_j^2}\right]^3 \tag{18}$$

with the shape parameter c_j kept constant as $1/N$. The shape parameter is used to control the smoothness of the RBF.

From Eq. 11 one can notice that a polynomial of order M is added to the radial basis function. M was limited to an upper value of 6. After inspecting Eqs. (11-14), one can easily check that the final linear system has $[(N+M^*L)+1]$ equations. Some tests were made using the cross-product polynomials $(x_i\, x_j\, x_k \ldots)$, but the improvements on the results were irrelevant. Also, other types of RBFs were used, but no improvement on the interpolation was observed.

To provide a more complete picture of metamodel accuracy, three different metrics are used: R Square, Relative Average Absolute Error (RAAE), and Relative Maximum Absolute Error (RMAE) [31].

a) R Square (R2)

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} = 1 - \frac{MSE}{\text{variance}} \tag{19}$$

where $\hat{y}_i$ is the corresponding predicted value for the observed value y_i; $\bar{y}$ is the mean of the observed values. While *MSE* (Mean Square Error) represents the departure of the metamodel from the real simulation model, the variance captures how irregular the problem is. The larger the value of R2, the more accurate the metamodel.

b) Relative Average Absolute Error (RAAE)

$$RAAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n * STD} \tag{20}$$

where *STD* stands for standard deviation. The smaller the value of RAAE, the more accurate the metamodel.

c) Relative Maximum Absolute Error (RMAE)

$$RMAE = \frac{\max(|y_1 - \hat{y}_1|, |y_2 - \hat{y}_2|, \ldots, |y_n - \hat{y}_n|)}{STD} \tag{21}$$

Large RMAE indicates large error in one region of the design space even though the overall accuracy indicated by R2 and RAAE can be very good. Therefore, a small RMAE is preferred. However, since this metric cannot show the overall performance in the design space, it is not as important as R2 and RAAE.

Although the R2, RAAE and RMAE are useful to ascertain the accuracy of the interpolation, they can fail in some cases. For the R2 metric, for example, if one of the testing points has a huge deviation of the exact value, such discrepancy might affect the entire sum appearing on Eq. (19) and, even if all the other testing points are accurately interpolated. Similarly, the R2 result can be very bad. For this reason, we also calculate the percentage deviation of the exact value of each testing point. Such deviations are collected according to 6 ranges of errors: 0-10%; 10-20%; 20-50%; 50-100%; 100-200%; >200%. Thus, a interpolation that has all testing points within the interval of 0 to 10% of relative error might be considered good in comparison to another one where the points are all spread along the intervals from 10% to 200%.

The procedure was shown to work for fitting highly non-linear functions where large number of variables were involved. The RBF technique is quite powerful regarding its accuracy and reduced CPU time. Even when the number of variables were as large as 500, the RBF approximation was very fast and robust. This is a promising technique for real time interpolations such as target tracking or image recognition. Some comparisons were made with the Wavelets Neural Network showing a general superior behavior of the RBF. A new hybrid optimizer based on the RBF interpolation was also presented, which is much more efficient than our previous ones. In fact, its performance is very close to IOSO software [21] which is one of the best commercial optimizers available.

Although the R2, RAAE and RMAE are useful to ascertain the accuracy of the interpolation, they can fail in some cases. For the R2 metric, for example, if one of the testing points has a huge deviation from the exact value, such discrepancy might affect the entire sum appearing in

Eq. (19) and, even if all the other testing points are accurately interpolated, the R2 result can be very bad. For this reason, we also calculate the percentage deviation from the exact value of each testing point.

In order to test the accuracy of the RBF models proposed, 295 test cases were used, representing linear and non-linear problems with up to 100 variables. Such problems were selected from a collection of 395 problems (actually 295 documented test cases), created by Hock and Schittkowski [32] and Schittkowski [33]. Figure 16 shows the number of variables of each one of the test cases. Note that there are 395 test cases, but some of them were not used because test cases ranging from 120 to 200 were not defined in the original publications.

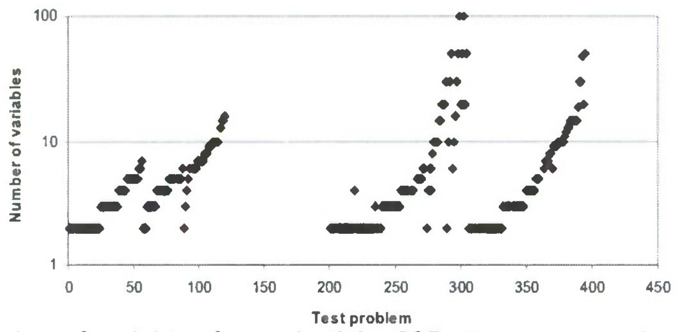


Figure 16. Number of variables for each of the 295 test cases considered for RBF fit.

In order to verify the accuracy of the interpolation over different number of training points, three sets were defined; scarce, small and medium. Also, the number of corresponding testing points varied according to the number of training points. Table 1 presents these three sets, based on the number of dimensions (variables) *L* of the particular problem.

Table 1. Number of training and testing points where L is the number of variables.

	Number of training points	Number of testing points
Scarce set	3 *L*	300 *L*
Small set	10 *L*	1000 *L*
Medium set	50 L	10000 L

For example, if we are trying to fit a function of say, 500 variables, then a scarce set will have 3*500 = 1500 training points and 300*500 = 150,000 testing points. The large matrix resulting from Eq. (11) assuming that we use polynomial of order M = 6, will be of size (6*500 + 1,500 + 1 = 4501) squared. Such large resulting matrices have been solved iteratively using bi-conjugate iterative method. Cross-validation was performed for each combination of these, say, 4501 x 4501 coefficients.

Figure 17 shows the R2 metric for all test cases, using the scarce set (3*L) of training points only for the interpolation functions that presented some meaningful results. It can be noticed that the results are all spread from 0 (completely inadequate interpolation) to 1 (very accurate multi-dimensional interpolation). Qualitatively, the darkest pictures mean better results. The fittest polynomial RBF method provides the best interpolation among all test functions. The RBF1, RBF4 and RBF7 based methods also performed a good interpolation, based on the R2 metric.

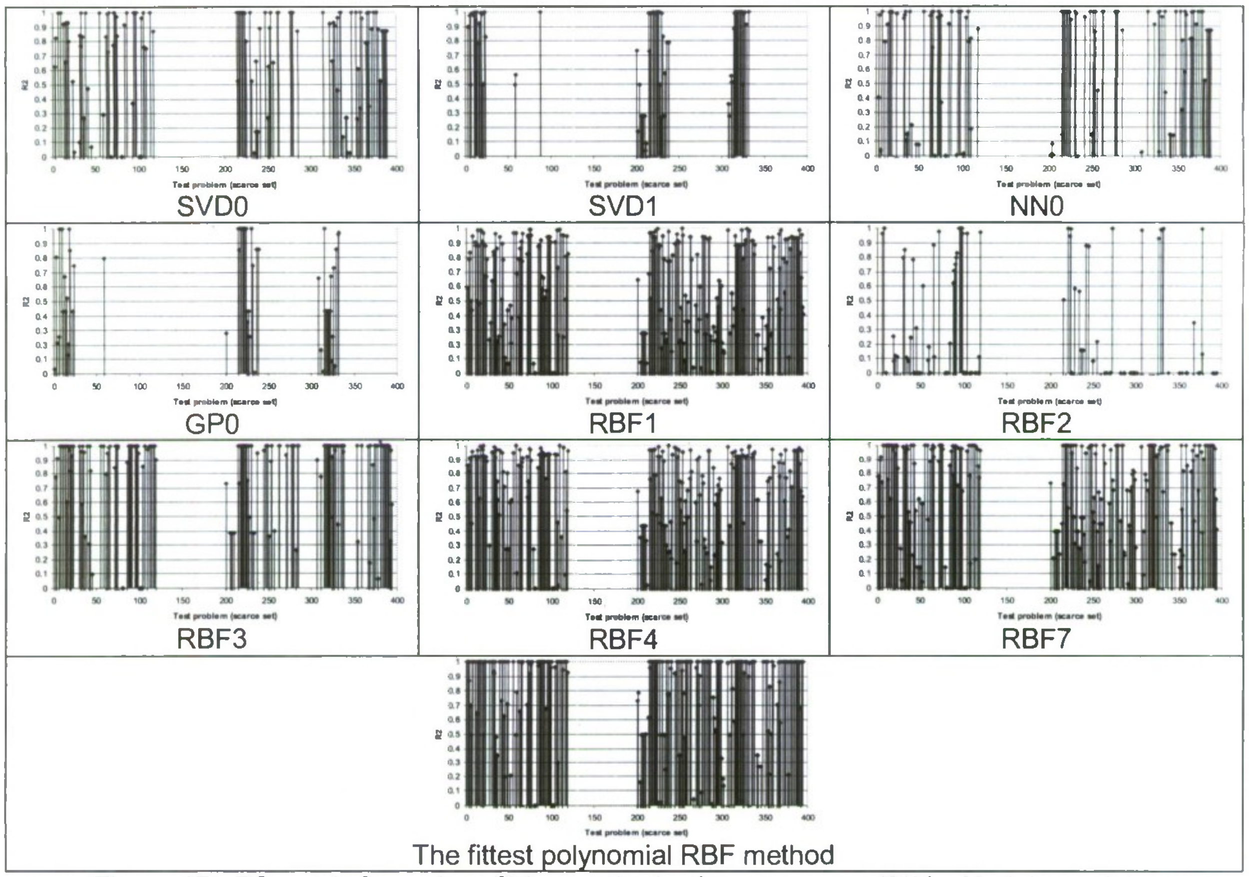


Figure 17. R2 metric for various fitting method using scarce set (3*L) of training points.

Figure 18 shows the computing time required to interpolate each test function, using the scarce set (3*L) of training points for the fittest polynomial RBF method. For most of the cases, the computing time was less than 10 seconds, using a Pentium IV 3 GHz with 1Gb RAM running Rocks 4.2.1 (Cydonia) and the Intel© ifort (IFORT) 9.1 20060323 compiler with the directives "-static-libcxa -static -O3 -r8". In fact, the highest dimensional test case, which has 100 variables required only 50 seconds to be interpolated.

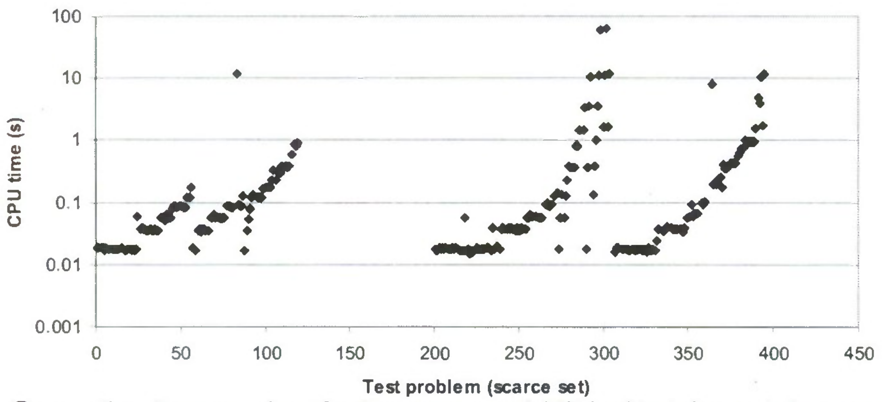


Figure 18. Computing time required for the scarce set (3*L) of training points using the fittest polynomial RBF method.

Response Surfaces Using a Hybridized Multi-Layer Self-Organizing

Self-organizing algorithms come from the field of cybernetics [34]. The idea is that this algorithm "learns" the black box model it is trying to mimic and makes the surrogate model only as complex as is required. By allowing the program to gradually complicate the final model, the construction and evaluation time of surrogate model is automatically optimized for a given task. In this multi-layer self-organizing algorithm very simple polynomial basis functions are used to generate models describing highly non-linear multi-variable functions.

Based on these concepts, we have developed and thoroughly tested a hybridized multi-layer self-organizing algorithm [35]. In this method, the algorithm can choose the basis functions that locally capture the interaction between the variables in the most accurate fashion. Two hybridized methods will be presented [35]. In the first method, the hybridized algorithm will be able to choose from linear, quadratic, cubic and quartic basis polynomials, as needed. In the second hybridized algorithm, the same basis polynomial options from the first hybrid method are given to the algorithm plus the capability to choose a simple RBF as a basis function to describe interactions among the variables. These methods will be compared to the single basis function response surface models using the same test problems.

This multi-layer self-organizing algorithms with different order polynomial basis functions will be compared using Schittkowski's suite of 295 non-linear optimization test cases [32, 33].

The self-organizing algorithm used in this work is the multilayer algorithm. In the multilayer algorithm the design variables are permutated, in pairs, to form nodes. At each node a least squares regression is performed using the two variables input to the nodes. These are variable vectors that are the size of the sample population. So, the output of the node is a vector of the predicted values from the regression

The polynomial used for the regression is a first order or second order polynomial. For instance, a second order basis polynomial would be:

$$\begin{aligned}
&y_{i,j}^{k,n} = a_0^{k,n} + a_1^{k,n}x_i^{k,n} + a_2^{k,n}x_j^{k,n} + a_3^{k,n}x_i^{k,n}x_j^{k,n} + a_4^{k,n}x_i^{k,n}x_i^{k,n} + a_5^{k,n}x_j^{k,n}x_j^{k,n}\\
&where\\
&i \;\; = 1,2,\mathrm{K} \text{ ,number of inputs to given layer}\\
&j \;\; = 1,2,\mathrm{K} \text{ ,number of inputs to given layer}\\
&k \qquad\qquad = \text{current layer}\\
&n \qquad = \text{node number at current layer}\\
&i \neq j
\end{aligned} \tag{22}$$

The output of the node is the vector predicted y values for the given input. The output of a node in layer *k*-1 becomes the input (provides an x_i vector) for layer *k*.

The notation in equation one is designed to inform the reader that the functions and polynomial coefficients pertain to a particular layer and particular node on the layer. The notation should also give the reader a feel for the computational resources needed to create and maintain a multilayer self-organizing model.

Figure 19 shows a possible multilayer surrogate model for a three variable engineering model. In the bottom layer (the zero layer) actual design variables are the nodes in the layer. These become the *x* inputs to layer 1. The nodes for layer 1 are created by permuting the input variables and performing least squares fit using Equation 22 and the actual response from the actual function (i.e., objective function, engineering simulation, etc.). Once layer 1 is made layer 2 is created but, now the nodes of layer 1 provide the *x*'s to make the new nodes using Equation 1. Now when layer 3 is to be made the 3rd node of layer 2 is not included. For now, we will just say that the results of that regression were not good enough to be used to make the 3rd layer. Since only two nodes from layer 2 were used to make layer three, only one node can be created in layer three and the process of making models ends there.

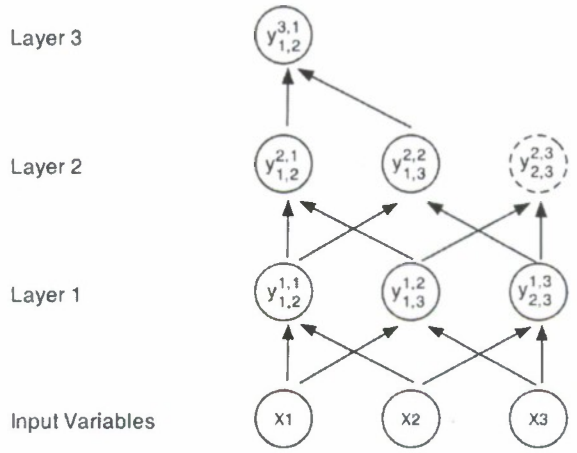


Figure 19. Example of a three-layer model for a multi-layer self-organizing algorithm.

The third node in layer 2 is not used to make nodes in layer 3 in Figure 19. A selection criterion is used to determine if the information in a node get passed on to the next layer. Madala and Ivakhnenko [34] suggest that using the following equation is an appropriate means for checking the quality of a node and can be a selection criterion.

$$\Delta^2(B) = \frac{\sum\limits_{p \in N_B} (y - \hat{y})_p^2}{\sum\limits_{p \in N_B} (y_p - \bar{y})^2} \tag{23}$$

Where :

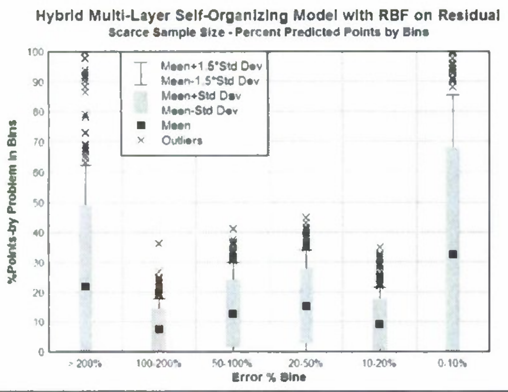
y_p = desired values

$\hat{y}_p$ = the predicted values

$\bar{y}$ = the mean of the desired values

In the multilayer algorithm a threshold is set for the maximum acceptable value of Eq. (23). Nodes that are within the threshold are passed on to the next layer. For each new layer the threshold is made smaller. This serves to minimize the amount of nodes in each layer to only the information that is needed to improve the network. This trimming of nodes is crucial to keeping the method compact and efficient.

The preliminary results for the hybrid model, polynomial basis only, combined with and RBF fit of the model residuals is presented for the 295 test cases of Schitkowski [32, 33].



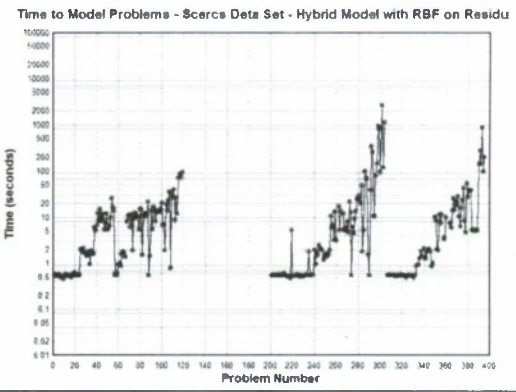
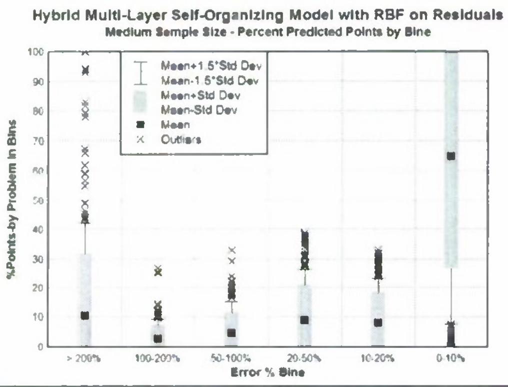


Figure 20 – Hybrid with RBF, Scarce Data Set, Model Accuracy	Figure 21 – Hybrid with RBF, Scarce Data Set, Time to Fit



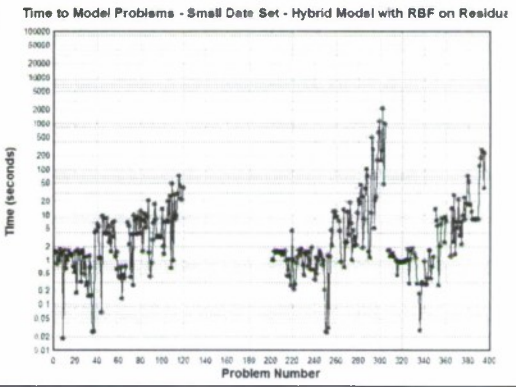
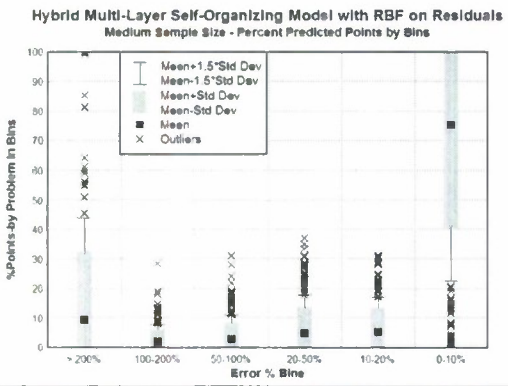


Figure 22 – Hybrid with RBF, Small Data Set, Model Accuracy	Figure 23 – Hybrid with RBF, Small Data Set, Time to Fit



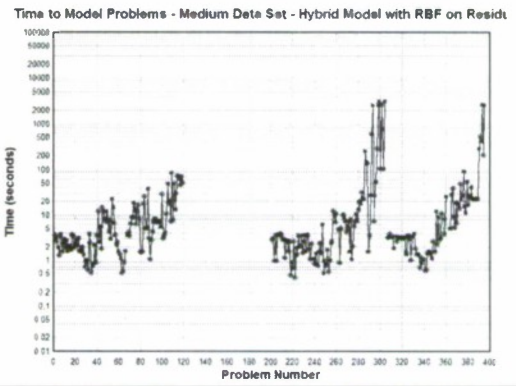


Figure 24 – Hybrid with RBF, Medium Size Data Set, Model Accuracy	Figure 25 – Hybrid with RBF, Medium Size Data Set, Time to Fit

The self-organizing multi-level fitting concept for multi-dimensional response surfaces appears to be suitable for very large dimensional response surfaces where the objective function depends on thousands of design variables. For smaller dimensionality response surfaces, this algorithm is too costly and cumbersome. The option of the multi-layer self-organizing algorithm that uses lower level (quadratic) local fits, and then uses a polynomial of radial basis functions to fit the errors of the quadratic local fits, offers the highest accuracy.

Uncertainty Evaluation using Bayesian Statistics

The algorithmic methods for the solution of inverse problems could be grouped into two basic approaches: pure inverse methods and optimization-based methods. That is, in most of the iterative methods for the solution of inverse problems, sophisticated regularization formulations are used in order to prevent numerical errors from growing exponentially. In many other methods, different optimization algorithms are used to solve *de facto* inverse problems by minimizing typically least-squares norms resulting from such algorithms. For an example, see Hernandez *et al.* [36]. A variety of techniques is currently available for the solution of inverse problems. However, one common approach relies on the minimization of an objective function that generally involves the squared difference between measured and estimated variables, such as the least-squares norm, as well as some kind of regularization term. Despite the fact that the minimization of the least-squares norm is indiscriminately used, it only yields *maximum likelihood* estimates if the following statistical hypotheses are valid:

a) the errors in the measured variables are additive, uncorrelated, normally distributed, with zero mean and known constant standard-deviation;
b) only the measured variables appearing in the objective function contain errors; and
c) there is no prior information regarding the values and uncertainties of the unknown parameters.

Although very popular and useful in many situations, the minimization of the least-squares norm is a non-Bayesian estimator. A Bayesian estimator is concerned with the analysis of the *posterior probability density*, which is the conditional probability of the parameters given the measurements, while the likelihood is the conditional probability of the measurements given the parameters.

If we assume the parameters and the measurements to be independent Gaussian random variables, with known means and covariance matrices, and that the measurement errors are additive, a closed form expression can be derived for the posterior probability density. In this case, the estimator that maximizes the posterior probability density can be recast in the form of a minimization problem involving the *maximum a posteriori objective function*.

On the other hand, if different *prior* probability densities are assumed for the parameters, the posterior probability distribution does not allow an analytical treatment. In this case, Markov Chain Monte Carlo (MCMC) methods are used to draw samples of all possible parameters, so that inference on the posterior probability becomes inference on the samples. As such, the number of samples required to accurately approximate the posterior distribution is generally large, resulting in prohibitive computational costs for many practical applications. Such is specially the case when the solution of the forward problem, which is needed for the computation of the likelihood function, requires large computational times.

In this work, we examined the use of radial basis functions to interpolate the likelihood function, in order to reduce the computational cost of MCMC methods in the Bayesian approach of solution of inverse problems. The likelihood function is interpolated in the space of all possible parameters, by using a small number of solutions of the forward model as compared to that required for the implementation of the MCMC methods. Hence, the interpolated likelihood function, instead of the actual function, is used afterwards in the sampling procedure of the MCMC method, providing a substantial reduction of computational costs.

Although very popular and useful in many situations, the minimization of the least-squares norm is a non-Bayesian estimator. A Bayesian estimator is concerned with the analysis of the *posterior probability density*, which is the conditional probability of the parameters given the measurements, while the likelihood is the conditional probability of the measurements given the parameters [37-40].

Markov Chain Monte Carlo (MCMC) Methods

In the Bayesian approach to statistics, an attempt is made to utilize all available information in order to reduce the amount of uncertainty present in an inferential or decision-making problem. As new information is obtained, it is combined with any previous information to form the basis for statistical procedures. The formal mechanism used to combine the new information with the previously available information is known as Bayes' theorem [40]. Therefore, the term *Bayesian* is often used to describe the so-called *statistical inversion approach*, which is based on the following principles [38]:

1. All variables included in the model are modeled as random variables.
2. The randomness describes the degree of information concerning their realizations.
3. The degree of information concerning these values is coded in probability distributions.
4. The solution of the inverse problem is the posterior probability distribution.

Consider, for the sake of generality, the vector of parameters appearing in the physical model formulation as

$$\mathbf{P}^T \equiv [P_1, P_2, \ldots, P_N] \tag{24}$$

and the vector of available measurements as

$$\mathbf{Y}^T \equiv [Y_1, Y_2, \ldots, Y_I] \tag{25}$$

where N is the number of parameters and I is the number of measurements. Bayes' theorem can then be stated as [38]:

$$\pi_{posterior}(\mathbf{P}) = \pi(\mathbf{P}|\mathbf{Y}) = \frac{\pi_{prior}(\mathbf{P})\pi(\mathbf{Y}|\mathbf{P})}{\pi(\mathbf{Y})} \tag{26}$$

where $\pi_{posterior}(\mathbf{P})$ is the posterior probability density, that is, the conditional probability of the parameters $\mathbf{P}$ given the measurements $\mathbf{Y}$; $\pi_{prior}(\mathbf{P})$ is the prior density, that is, the coded information about the parameters prior to the measurements; $\pi(\mathbf{Y}|\mathbf{P})$ is the likelihood function, which expresses the likelihood of different measurement outcomes $\mathbf{Y}$ with $\mathbf{P}$ given; and $\pi(\mathbf{Y})$ is the marginal probability density of the measurements, which plays the role of a normalizing constant.

In practice, such normalizing constant is difficult to compute and numerical techniques, such as Markov Chain Monte Carlo Methods, are required in order to obtain samples that accurately represent the posterior probability density. In order to implement the MCMC, a density function $q(\mathbf{P}^*, \mathbf{P}^{(t-1)})$ is required that gives the probability of moving from the current state in the chain $\mathbf{P}^{(t-1)}$ to a new state $\mathbf{P}^*$.

The Metropolis-Hastings algorithm was used in this work to implement the MCMC method. It can be summarized in the following steps [38-40]:

1. Sample a *Candidate Point* $\mathbf{P}^*$ from a jumping distribution $q(\mathbf{P}^*, \mathbf{P}^{(t-1)})$.
2. Calculate:

$$\alpha = \min\left[1, \frac{\pi(\mathbf{P}^* \mid \mathbf{Y})q(\mathbf{P}^{(t-1)}, \mathbf{P}^*)}{\pi(\mathbf{P}^{(t-1)} \mid \mathbf{Y})q(\mathbf{P}^*, \mathbf{P}^{(t-1)})}\right] \tag{27}$$

3. Generate a random value U that is uniformly distributed on (0,1).

4. If $U \le \alpha$ define $\mathbf{P}^t$=$\mathbf{P}^*$, otherwise, define $\mathbf{P}^t$= $\mathbf{P}^{(t-1)}$.

5. Return to step 1 in order to generate the sequence {$\mathbf{P}^1$, $\mathbf{P}^2$, ..., $\mathbf{P}^n$}.

In this way, we get a sequence that represents the posterior distribution and inference on this distribution is obtained from inference on the samples {$\mathbf{P}^1$, $\mathbf{P}^2$, ..., $\mathbf{P}^n$}. We note that values of $\mathbf{P}^i$ must be ignored until the chain has not converged to equilibrium. For more details on theoretical aspects of the Metropolis-Hastings algorithm and MCMC methods, the reader should consult references [38-40[.

We assume in this work that the errors in the measured variables are additive, uncorrelated, normally distributed, with zero mean and known constant standard deviation. Hence, the likelihood function is given by [37-40]

$$\pi(\mathbf{Y}|\mathbf{P}) = (2\pi)^{-I/2}\left|\mathbf{W}^{-1}\right|^{-1/2}\exp\left\{-\frac{1}{2}[\mathbf{Y} - \mathbf{T}(\mathbf{P})]^T\mathbf{W}[\mathbf{Y} - \mathbf{T}(\mathbf{P})]\right\} \tag{28}$$

where **T** is the vector of estimated variables, obtained from the solution of the forward model with an estimate for the parameters **P**, and **W** is the inverse of the covariance matrix of the measurements.

An example of application of this combined algorithmic approach is described below as applied to the inverse problem of estimating the three thermal conductivity components of an orthotropic solid [41-43].

As an example for the proposed research, the physical problem considered here involves the three-dimensional linear heat conduction in an orthotropic solid, with thermal conductivity components k_1^*, $k_2^*$and k_3^* in the x^*, y^* and z^* directions, respectively. The solid is a parallelepiped with sides a^*, b^* and c^*, initially at temperature T_0^*. For times $t^*> 0$, uniform heat fluxes $q_1^*(t)$, $q_2^*(t)$ and $q_3^*(t)$ are applied at the surfaces $x^* = a^*$, $y^* = b^*$ and $z^* = c^*$, respectively. The other three remaining surfaces at $x^* = 0$, $y^* = 0$ and $z^* = 0$ are maintained at a constant temperature (equal to the initial temperature). The mathematical formulation of this type of physical problem is given in *dimensionless form* by the following parabolic partial differential equation where T(x,y,z;t) is the unsteady three-dimensional temperature field:

$$k_1\frac{\partial^2 T}{\partial x^2} + k_2\frac{\partial^2 T}{\partial y^2} + k_3\frac{\partial^2 T}{\partial z^2} = \frac{\partial T}{\partial t} \quad \text{in } 0 < x < a, 0 < y < b, 0 < z < c\ ;\ t > 0 \tag{29.a}$$

$$T = 0 \text{ at } x = 0 \quad ; \quad k_1\frac{\partial T}{\partial x} = q_1(t) \text{ at } x = a \quad , \quad \text{for } t > 0 \tag{29.b,c}$$

$$T = 0 \text{ at } y = 0 \quad ; \quad k_2\frac{\partial T}{\partial y} = q_2(t) \text{ at } y = b \quad , \quad \text{for } t > 0 \tag{29.d,e}$$

$$T = 0 \text{ at } z = 0 \quad ; \quad k_3\frac{\partial T}{\partial z} = q_3(t) \text{ at } z = c \quad , \quad \text{for } t > 0 \tag{29.f,g}$$

$$T = 0 \quad \text{for } t = 0\ ;\ \text{in } 0 < x < a\ , 0 < y < b\ , 0 < z < c \tag{29.h}$$

In the *direct problem* associated with the physical problem described above, the three thermal conductivity components k_1, k_2 and k_3, as well as the solid geometry, initial and boundary conditions, are known. The objective of the direct problem is then to determine the transient temperature field $T(x, y, z, t)$ in the body.

We assume the boundary heat fluxes to be pulses of finite duration t_h, that is,

$$q_j(t) = \begin{cases} \bar{q}_j \quad , & \text{for} \quad 0 < t \le t_h \\ 0 \quad , & \text{for} \quad t > t_h \end{cases} \qquad \text{for } j = 1,2,3 \qquad (30)$$

where $\bar{q}_j$ is the *dimensionless* magnitude of the applied heat flux during the heating period $0 < t \le t_h$.

The solution of direct problem given by equation (1) for $0 < t \le t_h$ can be obtained with the Classical Integral Transform Technique as

$$T(x,y,z,t) = \frac{8}{a\,b\,c} \sum_{o=1}^{\infty} \sum_{n=1}^{\infty} \sum_{m=1}^{\infty} \Theta(\lambda_m, x)\, \Omega(\beta_n, y)\, \Psi(\gamma_o, z) \hat{\bar{T}}(\lambda_m, \beta_n, \gamma_o)\,(1 - e^{-t(k_1 \lambda_m^2 + k_2 \beta_n^2 + k_3 \gamma_o^2)}) \qquad (31)$$

where

$$\hat{\bar{T}}(\lambda_m, \beta_n, \gamma_o) = \frac{\dfrac{(-1)^{m+1} \bar{q}_1}{\beta_n \gamma_o} + \dfrac{(-1)^{n+1} \bar{q}_2}{\lambda_m \gamma_o} + \dfrac{(-1)^{o+1} \bar{q}_3}{\lambda_m \beta_n}}{k_1 \lambda_m^2 + k_2 \beta_n^2 + k_3 \gamma_o^2} \qquad (32)$$

and the eigenfunctions are given by

$$\begin{aligned} \Theta(\lambda_m, x) &= sin(\lambda_m\, x) \\ \Omega(\beta_n, y) &= sin(\beta_n\, y) \\ \Psi(\gamma_o, z) &= sin(\gamma_o\, z) \end{aligned} \qquad (33)$$

with eigenvalues

$$\lambda_m = \frac{(2\,m-1)}{2a}\pi\,; \quad \beta_n = \frac{(2\,n-1)}{2b}\pi\,; \quad \gamma_o = \frac{(2\,o-1)}{2c}\pi \qquad ; \; m,n,o = 1,2,\ldots \qquad (34)$$

For the *inverse problem* considered here, the thermal conductivity components k_1, k_2 and k_3 are regarded as unknown, while the other quantities appearing in the formulation of the direct problem described above are assumed to be known with high degree of accuracy. The *vector of unknown parameters* $\mathbf{P}^T = [k_1, k_2\,, k_3]$ is estimated by using the Bayesian technique described next.

Simulated temperature measurements, obtained with the solution of the direct problem with $m = n = o = 50$, were used in the inverse analysis. This number of terms in the series (4.a) was sufficient to achieve the desired convergence in the solution. In order to avoid the inverse crime of using the same solution of the direct problem in the generation of the simulated measurements and in the solution of the inverse problem [38], for the application of the Metropolis-Hastings algorithm we used $m = n = o = 20$ in the series-solution. The same optimally experimental variables selected in [41] were used here: for $k_1 = 1$, $k_2 = 15$ and $k_3 = 15$, we have $b/a = c/a = q_2/q_1 = q_3/q_1 = 3.87$ and the heating and final times were $t_h = t_f = 1$.

Results are presented below for the estimation of the thermal conductivity components $\mathbf{P}^T = [k_1, k_2\,, k_3]$ with the Metropolis - Hastings algorithm, by using 20000 samples. A uniform distribution was used as prior information for the thermal conductivity components. The unknowns were assumed to be in the broad physically meaningful intervals given by:

$$0.1 \leq k_1 \leq 50 \qquad 0.1 \leq k_2 \leq 50 \qquad 0.1 \leq k_3 \leq 50$$

Notice that the prior distributions assumed for the parameters are non-informative, that is, the intervals in which the parameters are uniformly distributed encompass most of engineering materials, ranging from mild insulators to metals. Therefore, this represents a very difficult test case for the solution of the inverse problem, especially when it is taken into account that thermal conductivity values used to generate the simulated measurements are quite different in the three directions.

Figures 26.a,b present the exact and estimated temperatures, as well as the simulated measurements, at the center of the heated surface at $x = a$, for standard deviations of the errors of $\sigma = 0.01$ and $\sigma = 0.05$, respectively. The estimated temperatures were obtained with the solution of the direct problem with the mean values estimated for the parameters. It should be noticed in figures 26.a,b that the estimated temperatures are in excellent agreement with the exact ones, even for very large measurement errors as for $\sigma = 0.05$ (see figure 26.b).

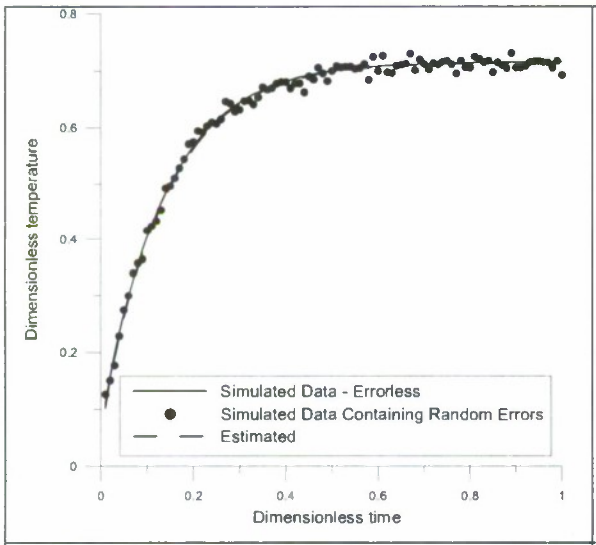


Figure 26.a – Temperature at the center of the surface at $x = a$. Simulated measurements with standard deviation $\sigma = 0.01$.

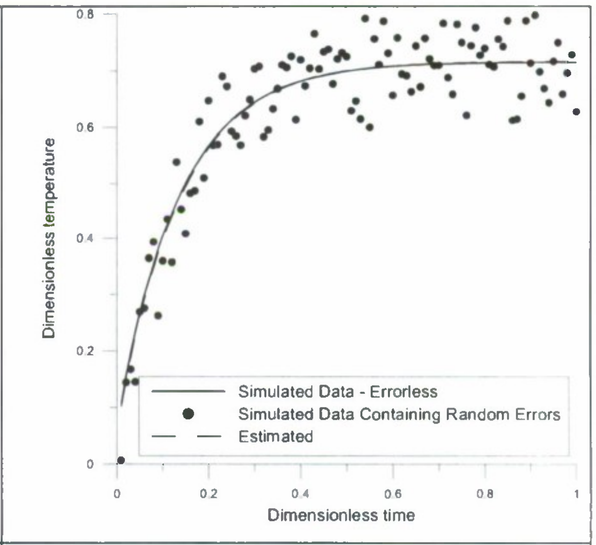


Figure 26.b – Temperature at the center of the surface at $x = a$. Simulated measurements with standard deviation $\sigma = 0.05$.

Table 2. Results obtained with the MCMC method

Standard Deviation for the measurements	Parameter	Mean	Standard Deviation
$\sigma = 0.01$	k_1	0.983	0.004
	k_2	14.74	0.06
	k_3	15.04	0.05
$\sigma = 0.05$	k_1	0.99	0.02
	k_2	14.8	0.2
	k_3	14.7	0.3

The mean values for the parameters, as well as their correspondent standard deviations were obtained from the posterior distribution generated with the Metropolis-Hastings algorithm (see table 2). This table shows that the MCMC Bayesian approach with the Metropolis-Hastings algorithm provided very accurate estimates for the unknown thermal conductivity components, even for very large magnitudes of the measurement errors, such as $\sigma = 0.05$. As expected, the standard deviations of the estimated parameters increase with the magnitude of the measurement errors.

The states of the Markov chain resulting from the application of the Metropolis-Hastings algorithm, for the cases with standard deviation of the measurement error of $\sigma = 0.05$ show that the posterior distribution for the parameters after the Markov chain reaches equilibrium resembles that for a linear estimation problem, i.e., an ellipsoid. Such is the fact despite the non-linearity of the present estimation problem and is due to the use of optimized experimental variables in the estimation procedure [41]. A burn-in period is required for the MCMC to reach equilibrium, where the samples come from their starting states to gradually form the posterior distribution for the parameters. As expected, the posterior distribution is more spread for larger measurement errors which is apparent from the analysis of figures 27, which present the states of the Markov chain for each parameter (marginal distributions) for $\sigma = 0.05$. The burn-in period for $\sigma = 0.01$ was of the order of 4000 states, while for $\sigma = 0.05$ the burn-in period was of the order of 2000 states. The mean values and the correspondent standard deviations shown in table 2 were computed by discarding the samples during the burn-in period.

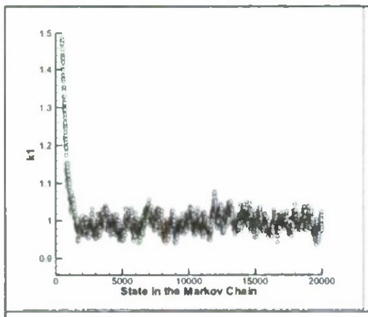
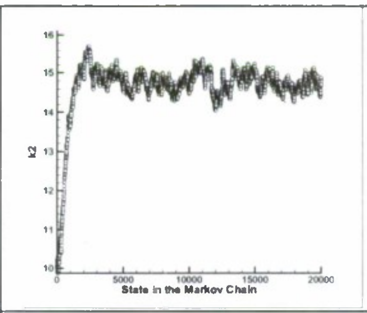
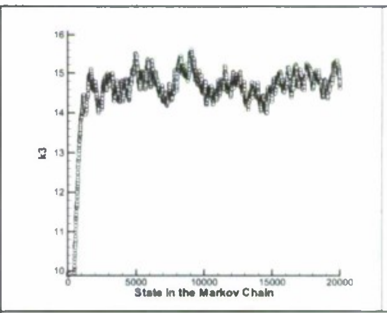
		
k_1	k_2	k_3

Figure 27 States of the Markov chain with standard deviation $\sigma = 0.05$ for three conductivities.

Although an order of magnitude computationally more expensive than Kalman filters, it is highly advisable to use MCMC with the Metropolis-Hastings algorithm especially when dealing with non-linear problems, since this approach offers higher accuracy and robustness.

List of Publications, Theses and Dissertations Resulting from this Project

Prof. Dulikravich and his graduate student assistants in collaboration with Prof. Marcelo J. Colaco and Prof. Helcio R. B. Orlande from Brazil, have produced the following publications that were either fully or in part supported by the funding from this grant.

Proceedings Published

1. Dulikravich, G. S., Colaco, M. J., Orlande, H. R. B. and Tanaka, M. (editors): *Inverse Problems, Design and Optimization (IPDO-2007) Vol. I*, ISBN: 978-1-59916-279-9, Florida International University, Miami, FL, June 2007.
2. Dulikravich, G. S., Colaco, M. J., Orlande, H. R. B. and Tanaka, M. (editors): *Inverse Problems, Design and Optimization (IPDO-2007) Vol. II*, ISBN: 978-1-59916-280-5, Florida International University, Miami, FL, June 2007.

Journal Articles Published

1. Approximation of the Likelihood Function in the Bayesian Technique for the Solution of Inverse Problems (with Orlande, H.R.B. and Colaco, M.J.), *Inverse Problems in Science and Engineering*, Vol. 16, issue 6, 2008, pp. 677-692
2. A Response Surface Method-Based Hybrid Optimizer (with Colaco, M. J. and Sahoo, D.), *Inverse Problems in Science and Engineering,* Vol. 16, issue 6, 2008, pp. 717-741.
3. Optimizing Chemistry of Bulk Metallic Glasses for Improved Thermal Stability (with Egorov, I. N. and Colaco, M. J.), *Modelling and Simulation in Materials Science and Engineering,* 16, 2008, 075010 (19pp).
4. Multi-Objective Hybrid Evolutionary Optimization Utilizing Automatic Algorithm Switching (with Moral, R. J.), *AIAA Journal*, Vol. 46, No. 3, March 2008, pp 673-700.
5. Identification and Design of a Source Term in a Two-Region Heat Conduction Problem (with Silva, P. P., Orlande, H. R. B., Colaco, M. J., Shiakolas, P. S.), *Inverse Problems in Science and Engineering*, Vol. 15, No. 7, 2007, pp. 661-677.
6. Multi-Objective Design Optimization of Topology and Performance of Branching Networks of Cooling Passages (with Gonzalez, M. J., Jelisavcic, N., Moral, R. J., Sahoo, D. and Martin, T. J. M.), *International Journal of Thermal Sciences*, Vol. 46, 2007, pp. 1191-1202.
7. Solidification of Double-Diffusive Flows Using Thermo-Magneto-Hydrodynamics and Optimization (with Colaco, M.J.), *Materials and Manufacturing Processes,* Vol. 22, 2007, pp. 594-606.
8. Inverse Approaches to Drying of Thin Bodies With Significant Shrinkage Effects (with G. H. Kanevce, L. P. Kanevce, V. B. Mitrevski, H.R.B. Orlande), *ASME Journal of Heat Transfer*, Vol. 129, March 2007, pp. 379-386.
9. A Multilevel Hybrid Optimization of Magnetohydrodynamic Problems in Double-Diffusive Fluid Flow (with Colaco, M. J.), *Journal of Physics and Chemistry of Solids*, Vol. 67, 2006, pp. 1965-1972.

Conference Papers Published

1. Inverse Design of Topology of Branching Passages for Viscous Flows (with Wood, S.), paper IMECE2009-10740, 2009 ASME IMECE Congress, Orlando, FL, November 15-19, 2009.
2. Optimizing Concentrations of Alloying Elements and Tempering of Corrosion Resistant Aluminum Alloys (with Bhargava, S., Colaco, M. J. and Murty, S.), 10th US Congress of Computational Mechanics, Columbus, OH, July 10-15, 2009.
3. Modified Predator-Prey (MPP) Algorithm for Constrained Multi-objective Optimization (with Chowdhury, S. and Moral, R. J.), EUROGEN 2009, Cracow, Poland, June 15-17, 2009.

4. Development of Digital Filters for Inverse Heat Conduction Problems (with de Sousa, P., Fernandes, A. P., Guimaraes, G.), 12th Brazilian Congress of Thermal Engineering and Sciences, Belo Horizonte, MG, Brazil, November 10-14, 2008.
5. A Hybrid Self-Organizing Response Surface Methodology (with Moral, R. J.), paper AIAA-2008-5891, 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Victoria, British Columbia, Canada, September 10-12, 2008.
6. Bayesian Estimation of the Thermal Conductivity Components of Orthotropic Solids (with Orlande, H. R. B. and Colaco, M. J.), 5th Brazilian National Congress of Mechanical Engineering - CONEM, Salvador, Bahia, Brazil, August 18-21, 2008.
7. Identification of Parameters in a System of Differential Equations Modeling Evolution of Infectious Diseases (with Hernandez, A., Colaco, M. J. and Moral, R. J.), ASME paper DETC2008-49595, Symposium at the 2008 ASME International Design Engineering Technical Conferences (IDETC) and Computers and Information in Engineering Conference (CIE), (eds. Dulikravich, G.S., Michopoulos, J. and Kumar, A. V.), Brooklyn, NY, August 3-6, 2008.
8. Predator-Prey Evolutionary Multi-Objective Optimization Algorithm: Performance and Improvements (with Chowdhury, S. and Moral, R. J.), 7th conference of Association for Structural and Multidisciplinary Optimization in the UK (ed: Toropov, V. V.), Bath, U.K., July 7-8, 2008.
9. A Comparison of Accuracy and Robustness of Methods for Construction of Multi-dimensional Response Surfaces (with Colaco, M. J.), Minisymposium on "Metamodels for High Dimensionality Response Surfaces in Multiobjective Optimization", 8th World Congress of Computational Mechanics, (eds: Dulikravich, G. S. and Colaco, M. J.), Venice, Italy, June 30-July 5, 2008.
10. Application of Bayesian Filters to Heat Conduction Problems (with Orlande, H. R. B. and Colaco, M. J.), EngOpt 2008 - International Conference on Engineering Optimization, (ed: Herskovits, J.), Rio de Janeiro, Brazil, June 1-5, 2008.
11. A Response Surface Method Based Hybrid Optimizer (with Colaco, J. M.), EngOpt 2008 - International Conference on Engineering Optimization, (ed: Herskovits, J.), Rio de Janeiro, Brazil, June 1-5, 2008.
12. Inverse Estimation of Moisture Diffusivity by Utilizing Temperature Response of a Drying Body (with G. Kanevce, L. Kanevce, V. Mitrevski), ICCES'08: International Conference on Computational & Experimental Engineering and Sciences, Honolulu, Hawaii, March 17-22, 2008.
13. Electro-Thermo-Hydrodynamic Control of Solidification of Binary Mixtures (with Colaco, M. J.), 9th International Symposium on Fluid Control, Measurement and Visualization – FLUCOME 2007, ed. Chen, C.-Y., Florida State University, Tallahassee, FL, Sept. 16-19, 2007.
14. Approximation of the Likelihood Function in the Bayesian Technique for the Solution of Inverse Problems (with Orlande, H.R.B. and Colaco, M.J.), International Symposium on Inverse Problems, Design and Optimization (IPDO-2007), (eds.: Dulikravich, G.S., Orlande, H.R.B., Tanaka, M. and Colaco, M.J.), Miami Beach, FL, April 16-18, 2007.
15. A Comparison of Two Methods for Fitting High Dimensional Response Surfaces (with Colaco, M. J. and Sahoo, D.), International Symposium on Inverse Problems, Design and Optimization (IPDO-2007), (eds.: Dulikravich, G.S., Orlande, H.R.B., Tanaka, M. and Colaco, M.J.), Miami Beach, FL, April 16-18, 2007.
16. Inverse Approaches to Drying of Sliced Foods (with Kanevce, G. H., Kanevce, Lj. P., and Mitrevski, V. B.), International Symposium on Inverse Problems, Design and Optimization (IPDO-2007), (eds.: Dulikravich, G.S., Orlande, H.R.B., Tanaka, M. and Colaco, M.J.), Miami Beach, FL, April 16-18, 2007.
17. Inverse Approaches in Improvement of Air Pollution Plume Dispersion Models for

Regulatory Applications (with Kanevce, G. H., Kanevce, Lj. P., and Andreevski, I. B.), International Symposium on Inverse Problems, Design and Optimization (IPDO-2007), (eds.: Dulikravich, G.S., Orlande, H.R.B., Tanaka, M. and Colaco, M.J.), Miami Beach, FL, April 16-18, 2007.

18. *Controlling Solidification of Particle Laden Melts Using Thermo-Magnetohydrodynamics and Optimization (with Colaco, M. J.), Symposium on Processing for Reliability, (ed: Mullins, W. M.), Materials Science & Technology 2006, Cincinnati, OH, Oct. 15-19, 2006.
19. *Evolutionary Wavelet Neural Network for Large Scale Function Estimation in Optimization (with Sahoo, D.), AIAA Paper AIAA-2006-6955, 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Portsmouth, VA, September 6-8, 2006.
20. *Multi-Objective Hybrid Evolutionary Optimization With Automatic Switching (with Moral, R. J. and Sahoo, D.), AIAA Paper AIAA-2006-6976, 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Portsmouth, VA, September 6-8, 2006.
21. Inverse Approaches to Drying With and Without Shrinkage (with Kanevce, G., Kanevce, Lj., Mitrevski, V. and Orlande, H. R. B.), Proceedings of 15th International Drying Symposium (IDS 2006), (eds: Farkas, I.), Vol. A, pp. 576-584, Budapest, Hungary, Aug. 20-23, 2006.

Publications (accepted or submitted), but not yet published during the reporting period:

1. Improvements to Single-Objective Constrained Predator-Prey Evolutionary Optimization Algorithm (with Chowdhury, S.), *Structural and Multidisciplinary Optimization*.
2. Robustness, Accuracy and Efficiency of Different Response Surface Methods (with Colaco, M J., da Silva, W. B., and Magalhaes, A. C.), *AIAA Journal*.
3. Modified Predator-Prey Algorithm for Constrained and Unconstrained Multi-objective Optimization Problems (with Chowdhury, S. and Moral. R. J.), *IEEE Transactions on Evolutionary Computation*.
4. Bayesian Estimation of the Thermal Conductivity Components of Orthotropic Solids (with Orlande, H. R. B. and Colaco, M. J.*), High Temperatures - High Pressures; International Journal of Thermophysical Properties Research*.
5. Evolutionary Wavelet Neural Network for Multidimensional Function Estimation in Optimization (with Sahoo, D.), *Optimization Methods and Software*.

Ph.D. and M.Sc. Thesis Awarded During these Three Years:

Ramon J. Moral: "Hybrid Multi-Objective Optimization and Hybridized Self-Organizing Response Surface Method", Ph.D. dissertation, Dept. of Mechanical and Materials Eng., Florida International University, Miami, FL, August 2008.

Souma Chowdhury: "Modified Predator-Prey (MPP) Algorithm for Constrained Single –and Multi-Objective Optimization Problems", M.Sc thesis, Dept. of Mechanical and Materials Eng., Florida International University, Miami, FL, December 2008.

Alexandre Aidov: "Modified Continuous Ant Colony Algorithm for Function Optimization", Dept. of Mechanical and Materials Eng., Florida International University, Miami, FL, August 2008.

M.Sc. Thesis in Progress

Ricardo Ardila: "Multi-Objective Optimization of Topology and Performance of Three-dimensional Networks of Cooling Passages", M.Sc. degree in Mechanical Engineering, Florida International University, Miami, FL, expected August 2009.

Carlos Velez: "Comparative Multi-Objective Composition Optimization and Experimental Evaluation of Bulk Metallic Glasses", M.Sc. degree in Mechanical Engineering, Florida International University, Miami, FL, expected August 2009.

Suvrat Bhargava: "Self Organizing Maps: Application to Multi-objective Optimization of Corrosion Resistant Aluminum Alloys", M.Sc. degree in Mechanical Engineering, Florida International University, Miami, FL, expected April 2010.

References

1. Dulikravich, G. S., Moral, R. J. and Sahoo, D., "A Multi-Objective Evolutionary Hybrid Optimizer", EUROGEN 05 - Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems, (eds: R. Schilling, W. Haase, J. Periaux, H. Baier, G. Bugeda), FLM, Munich, Germany, September 12-14, 2005.
2. Moral, R. J., Sahoo, D. and Dulikravich, G. S., "Multi-Objective Hybrid Evolutionary Optimization With Automatic Switching", AIAA Paper AIAA-2006-6976, 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Portsmouth, VA, September 6-8, 2006.
3. Moral, R. J. and Dulikravich, G. S., "Multi-Objective Hybrid Evolutionary Optimization Utilizing Automatic Algorithm Switching", *AIAA Journal*, Vol. 46, No. 3, 2008, pp 673-700, 2008.
4. Talbi, E.-G., "A Taxonomy on Hybrid Metaheuristics", *Journal of Heuristics*, Vol. 8, pp. 541-562, 2002.
5. Zitzler, E., Laumanns, M. and Thiele, L., "SPEA2: Improving the Strength Pareto Evolutionary Algorithm", TIK-Report 103, Dept. of Electrical Engineering, Swiss Federal Institute of Technology (ETH) Zurich, pp.1-21, 2001.
6. Parsopoulos, K. and Vrahatis, M., "Particle Swarm Optimization Method in Multi-objective Problems", Proceedings of the 2002 ACM Symposium on Applied Computing (SAC), 2002, pp. 603-607.
7. Eberhardt, R., Shi, Y. and Kennedy, J., *Swarm Intelligence*, Morgan Kaufmann, 2001.
8. Deb, K., Pratap, A., Agarwal, S. and Meyrivan, T., "A Fast and Elitist Multi-objective Genetic Algorithm: NSGA-II", *IEEE Trans. on Evolutionary Computation*, Vol. 6, No. 2, pp. 182-197, 2002.
9. Storn, R. and Price, K. V., "Minimizing the Real Function of the ICEC'96 Contest by Differential Evolution", IEEE Conference on Evolutionary Computation, pp. 842-844, 1996.
10. Zitzler, E., Thiele, L. and Deb, K., "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results", *Evolutionary Computation*, Vol. 8, No. 2, M.I.T. Press, pp. 173-195, 2000.
11. Zitzler, E., Thiele, L., Fonseca, C. M. and Grunert da Fonseca, V., "Performance Assessment of Multiobjective Optimizers: An Analysis and Review", *IEEE Transactions on Evolutionary Computations*, Vol. 7, No. 2, IEEE, 2003.
12. Deb, K., Multi-Objective Optimization Using Evolutionary Algorithms, John Wiley & Sons. 2002.
13. Sobol, I. M., "Uniformly Distributed Sequences with an Additional Uniform Property", *USSR Computational Mathematics and Mathematical Physics*, Vol. 16, 1976, pp. 236-242.
14. Bratley, P. and Fox, B., "ALGORITHM 659: Implementing Sobol's Quasirandom Sequence Generator", *ACM Transactions on Mathematical Software*, Vol. 14, Issue 1, March 1988.
15. Vrugt, J. A., and Robinson, B. A., "Improved Evolutionary Optimization from Genetically Adaptive Multimethod Search", Proceeding of the National Academy of Science (U.S.), Vol. 104, No. 3, pp. 708-711, 2007.
16. Colaço, M. J., Dulikravich, G. S., Orlande, H.R.B. and Martin, T. J., "Hybrid optimization with automatic switching among optimization algorithms", in: *Handbooks on Theory and Engineering Applications of Computational Methods: Evolutionary Algorithms and its Applications*, eds. Onate, Annicchiarico, Periaux, Barcelona, Spain, CIMNE, 2005.
17. Colaco, M. J. and Dulikravich, G. S., "Controlling Solidification of Particle Laden Melts Using Thermo-Magnetohydrodynamics and Optimization", Symposium on Processing for Reliability, (ed: Mullins, W. M.), Materials Science & Technology 2006, Cincinnati, OH, October 15-19, 2006.

18. Colaco, M. J. and Dulikravich, G. S., "A Multilevel Hybrid Optimization of Magneto-hydrodynamic Problems in Double-Diffusive Fluid Flow", *Journal of Physics and Chemistry of Solids*, Vol. 67, 2006, pp. 1965-1972.
19. Colaco, M. J. and Dulikravich, G. S., "Solidification of Double-Diffusive Flows Using Thermo-Magneto-Hydrodynamics and Optimization", *Materials and Manufacturing Processes*, Vol. 22, 2007, pp. 594-606.
20. More, J.J., Gabow, B.S. and Hillstrom, K.E., "Testing Unconstrained Optimization Software", ACM Trans. Meth. Software, 1981, pp. 17-41.
21. IOSO NM Version 1.0, User's Guide, IOSO Technology Center, Moscow, Russia, 2003.
22. Sandgren, E., "The Utility of Nonlinear Programming Algorithms", Ph.D. thesis, Purdue University, Indiana, USA, 1977.
23. Miele, A. and Cantrell, J.W., "Study on a memory gradient method for the minimization of functions", J. Optim. Theory and Applics., vol. .3, no. 6, 1969, pp. 459-470.
24. Sahoo, D. and Dulikravich, G. S., "Evolutionary Wavelet Neural Network for Large Scale Function Estimation in Optimization", *11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, AIAA Paper AIAA-2006-6955, Portsmouth, VA, September 6-8, 2006.
25. Colaço, M. J., Dulikravich, G. S. and Sahoo, D., "A Comparison of Two Methods for Fitting High Dimensional Response Surfaces", *Proc. of International Symposium on Inverse Problems, Design and Optimization (IPDO-2007)*, (eds.: Dulikravich, G. S., Orlande, H. R. B., Tanaka, M. and Colaco, M. J.), Miami Beach, Florida, U.S.A., April 16-18, 2007.
26. Colaço, M. J., Silva, W. B., Magalhães, A. C. and Dulikravich, G. S., "Response Surface Methods Applied to Scarce and Small Sets of Training Points – A Comparative Study", *EngOpt 2008 - International Conference on Engineering Optimization*, (ed: Herskovits, J.), Rio de Janeiro, Brazil, June 1-5, 2008.
27. Colaco, M. J. and Dulikravich, G. S., "A Response Surface Method Based Hybrid Optimizer", EngOpt 2008 - International Conference on Engineering Optimization, (ed: Herskovits, J.), Rio de Janeiro, Brazil, June 1-5, 2008.
28. Colaco, M. J. and Dulikravich, G. S., "RBF-Based Methods for Highly Multidimensional Response Surfaces Using Scarce Data Sets", paper AIAA-2008-5892, 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Victoria, British Columbia, Canada, September 10-12, 2008.
29. Buhmann, M.D., "Radial Basis Functions on Grids and Beyond", International Workshop on Meshfree Methods, Lisbon, Portugal, 2003.
30. Wendland, H., "Error Estimates for Interpolation by Compactly Supported Radial Basis Functions of Minimal Degree", *Journal of Approximation Theory*, Vol. 93, 1996, pp. 258-272.
31. Jin, R., Chen, W. and Simpson, T.W., "Comparative Studies of Metamodeling Techniques under Multiple Modeling Criteria", Proceedings of the 8th AIAA / USAF / NASA / ISSMO Multidisciplinary Analysis & Optimization Symposium, AIAA 2000-4801, Long Beach, CA, September 6-8, 2000.
32. Hock, W. and Schittkowski, K., "Test Examples for Nonlinear Programming Codes", Lecture Notes in Economics and Mathematical Systems, Vol. 187, Berlin / Heidelberg / New York: Springer-Verlag, 1981.
33. Schittkowski, K., "More Test Examples for Nonlinear Programming", Lecture Notes in Economics and Mathematical Systems, Vol. 282, Springer Verlag, 1987.
34. Madala, H.R. and Ivakhnenko, A.G., 1994, "Inductive Learning Algorithms for Complex Systems Modeling", CRC Press, Boca Raton, Florida, USA.
35. Moral, R. J. and Dulikravich, G. S., "A Hybrid Self-Organizing Response Surface Methodology", *paper AIAA-2008-5891, 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Victoria, British Columbia, Canada, September 10-12, 2008.

36. Hernandez, A., Dulikravich, G. S., Blower, S., Colaco, M. J. and Moral, R. J., "Identification of Parameters in a System of Differential Equations Modeling Evolution of Infectious Diseases", ASME paper DETC2008-49595, 2008 ASME International Design Eng. Technical Conferences (IDETC) and Computers and Information in Eng. Conference (CIE), New York, NY, August 3-6, 2008.
37. Beck, J. V. and Arnold, K., *Parameter Estimation in Engineering and Science*, Wiley Interscience, New York. 1977.
38. Kaipio, K. and Somersalo, E., *Statistical and Computational Inverse Problems*, Applied Mathematical Sciences 160, Springer-Verlag, 2004.
39. Tan, S., Fox, C., Nicholls, G., *Inverse Problems*, Course Notes for Physics 707, University of Auckland, New Zealand, 2006.
40. Lee, P., *Bayesian Statistics*, Oxford University Press, London, 2004.
41. Mejias; M. M., Orlande, H. R. B., Ozisik, M. N., "Effects of the Heating Process and Body Dimensions on the Estimation of the Thermal Conductivity Components of Orthotropic Solids". *Inverse Problems in Engineering*, Vol. no. 11, pp. 75-89, 2003.
42. Orlande, H. R. B., Colaco, M. J. and Dulikravich, G. S., "Response Surface Approximation for the Reduction of Computational Cost of Particle Filters", *EngOpt 2008 - International Conference on Engineering Optimization*, (ed: Herskovits, J.), Rio de Janeiro, Brazil, June 1-5, 2008.
43. Orlande, H. R. B., Colaco, M. J. and Dulikravich, G. S., "Bayesian Estimation of the Thermal Conductivity Components of Orthotropic Solids", *5th Brazilian National Congress of Mechanical Engineering - CONEM*, Salvador, Bahia, Brazil, August 18-21, 2008.